UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

EDISON PIGNATON DE FREITAS

Cooperative Context-Aware Setup and Performance of Surveillance Missions Using Static and Mobile Wireless Sensor Networks

Thesis presented in partial fulfilment
of the requirements for the degree of
Doctor of Computer Science

Prof. Dr. Carlos Eduardo Pereira
Advisor

Prof. Dr. Flávio Rech Wagner
Co-Advisor

Prof. Dr. Tony Larsson
Co-Advisor

Porto Alegre, Dezembro de 2011

CIP – CATALOGING-IN-PUBLICATION

Freitas, Edison Pignaton de

Cooperative Context-Aware Setup and Performance of Surveillance Missions Using Static and Mobile Wireless Sensor Networks. – Porto Alegre: PPGC da UFRGS, 2011.

299 p.: il.

Thesis (Ph.D.) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, BR– RS, 2011. Supervisor: Carlos Eduardo Pereira; Cosupervisor: Flávio Rech Wagner; Cosupervisor: Tony Larsson.

1. Surveillance systems. 2. Wireless sensor network. 3. Cooperative sensors. 4. Mobile sensors. 5. Biologically-inspired networking. 6. Context awareness. I. Pereira, Carlos Eduardo. II. Rech Wagner, Flávio. III. Larsson, Tony. IV. Título.

To my parents, Dílvia and Edisson, for all their love and support.

ACKNOWLEDGEMENTS

First I want to thank God for illuminating and protecting my life.

This thesis is an important milestone in my research career. Its development provided me the opportunity to take an important step towards maturation as researcher. Many people have helped me in several ways to achieve it, but first I would like to thank my main supervisors, Professors Carlos Eduardo Pereira and Tony Larsson for their valuable orientation and guidance since the beginning of my graduate studies. Special thanks for Professor Flávio Rech Wagner, my co-supervisor at UFRGS, which played a very important role in the development of my graduate studies. I would like to express my gratitude to these three professors that put a lot of effort in helping me to develop this thesis during the four years of my doctoral studies.

Thanks to Professor Antanas Verikas at Halmstad University and Mathias Broxvall at Örebro University for their feedback and comments, which were a valuable to improve the thesis. Thanks a lot to Professors Cláudio Fernando Resin Geyer and Luciano Paschoal Gaspary at UFRGS and Professor Antônio Alfredo Ferreira Loureiro at UFMG for their contribution during the defense of my thesis proposal. Your feedback, comments and suggestions were very important to improve my work since the defense of my thesis proposal. Thanks a lot!

Special thanks to Lieutenant-Colonel Dr. Armando Morado Ferreira, first for all his help in making possible the continuation of my studies in a graduate program linked with strategic interests of the Brazilian Army, as well as for his continued support during my graduate studies. Thanks to Lieutenant-Colonel Jorgito Matiuzzi Stocchero for believing in the potential of this work and supporting it within the Brazilian Army.

Many thanks to my PhD student colleagues that have been working with me in the SUAVIC (Sensors and Unmanned Aerial Vehicles Integration and Coordination) project, from which this thesis work is part of, Rodrigo Schmidt Allgayer, Ivan Muller and Andre Cavalcante at UFRGS. Special thanks for Professor Tales Heimfarth at Federal University of Lavras (UFLA), Brazil, for all the help and co-work in my graduate studies, especially in relation to the experiments presented in this work. Thanks also to Bachelor in Informatics Ivayr Farah Netto and Bachelor students at UFLA Alex Guimarães Cardoso de Sá and Luiz Augusto Guimarães Costa for implementing many parts of the performed simulations, and to MSc student at Halmstad University Bernhard Bösch for implementing the software used in the demonstrator.

Thanks to my colleagues at Halmstad University, Wagner, Yan, Zain, Annette and Kristoffer for their friendship and discussions about several issues related to our studies and also about life in general. I would like to extend my thanks to all members of CERES at Halmstad University for the nice working environment that we share daily.

Thanks to the administration staff at IDE department in Halmstad University, especially Eva Nestius and Jessika Rosenberg, for the support related to material acquisition, conference registrations and travel support. Thanks to secretary staff at UFRGS for all administrative support related to my graduate studies, especially to Janice Maisner de Oliveira for helping me with several local issues at UFRGS while I was abroad.

Finally, but definitely not less important, I would like to thank my family for all support and patience.

SUMMARY

ACRONYMS

ACL: Agent Communication Language
ACO: Ant Colony Optimization

BDI: Belief Desire Intention

COTS: Commercial Of The Shelf

DAI: Distributed Artificial Intelligence
DARPA: Defence Advanced Research Projects Agency – an agency of the United States government
DTN: Delay Tolerant Network

FIPA: Foundation for Intelligent Physical Agents
FSM: Finite State Machine

GPS: Global Positioning System
GUI: Graphical User Interface

HW: Hardware

IDE: Integrated Development Environment
IEEE: Institute of Electrical and Electronics Engineers
IR: Infra Red
ISAR: Inverse Synthetic Aperture Radar
ISR: Intelligence Surveillance and Reconnaissance

LAN: Local Area Network

MAC: Media Access Control
MAS: Multi-Agent Systems
MRTA: Multi-Robot Task Allocation
MWSN: Mobile Wireless Sensor Network

nesC: Network Embedded Systems C

OAP: Optimal Assignment Problem
QoS: Quality of Service

RADAR: Radio Detection and Ranging
RFID: Radio-frequency identification
RSSI: Received Signal Strength Indicator

SANET: Sensor/Actuator Network
SAR: Synthetic Aperture Radar
SW: Software
SQL: Structural Query Language

UAV: Unmanned Aerial Vehicle
UGV: Unmanned Ground Vehicle
UML: Unified Modeling Language
UUV: Under water Unmanned Vehicle

VANET: Vehicular Ad Hoc Network
VSN: Vehicular Sensor Network

WSN: Wireless Sensor Network

XML: Extensible Markup Language

LIST OF FIGURES

LIST OF TABLES

LIST OF LISTINGS

ABSTRACT

Surveillance systems are usually employed to monitor wide areas in which their users are interested in detecting and/or observing events or phenomena of their interest. The use of wireless sensor networks in such systems is of particular interest as these networks can provide a relative low cost and robust solution to cover large areas. Emerging applications in this context are proposing the use of wireless sensor networks composed of both static and mobile sensor nodes. Motivation for this trend is to reduce deployment and operating costs, besides providing enhanced functionalities.

This work focuses on the proposal of solutions for wireless sensor networks including static and mobile sensor nodes specifically regarding cooperative and context aware mission setup and performance. The goal is to keep the communication costs as low as possible in the execution of the proposed solutions. This concern comes from the fact that communication increases energy consumption, which is a particular issue for energy constrained sensor nodes often used in wireless sensor networks, especially if battery supplied. In the case of the mobile nodes, this energy constraint may not be valid, since their motion might need much more energy, but links instabilities and short time windows available to receive and transmit data. Therefore, it is better to communicate as little as possible.

For the interaction among static sensor nodes, the problems of dissemination and allocation of sensing missions are studied and a solution that explores local information is proposed and evaluated. This solution uses mobile software agents that have capabilities to take autonomous decisions about the mission dissemination and allocation using local context information. For mobile wireless sensor networks, the problem studied is how to perform handover of missions among the nodes according to their movements and locations in relation to the place where the missions have to be performed. To handle this problem, a mobile agent approach is proposed in which the agents implement the sensing missions' migration from node to node using geographical context information to decide about their migrations. For the networks combining static and mobile sensor nodes, the cooperation among them is approached by a biologically-inspired mechanism to deliver data from the static to the mobile nodes. The data delivery mechanism explores an analogy based on the behaviour of ants building and following trails, inspired by the ant colony algorithm.

The proposed solutions are flexible, being able to be applied to different application domains. Obtained experimental results provide evidence of the scalability of these proposed solutions, for example by evaluating their cost in terms of communication, among other metrics of interest for each solution. These results are compared to those achieved by reference solutions (theoretical optimum and flooding-based), providing indications of the proposed solutions' efficiency. These results are considered close to the theoretical optimum one and significantly better than the ones achieved by flooding-based solutions.

Keywords: Surveillance systems; Wireless sensor network; Cooperative sensors; Mobile sensors; Biologically-inspired networking; Context awareness.

RESUMO

Sistemas de vigilância são geralmente empregados no monitoramento de áreas de grandes dimensões nas quais seus usuários visam detectar ou observar fenômenos de seu interesse. O uso de redes de sensores sem fio nesses sistemas apresenta especial interesse, uma vez que essas redes podem apresentar soluções de baixo custo e robustas para cobrir áreas extensas. Neste contexto, novas aplicações têm surgido propondo o uso de redes de sensores sem fio compostas por nós sensores estáticos e móveis. Uma das motivações para esta tendência é a redução do custo de implantação e operação do sistema, além da possibilidade de proporcionar incremento em suas funcionalidades.

O foco desta tese se concentra na proposta de soluções para redes de sensores sem fio com uso cooperativo de sensores estáticos e móveis, com particular atenção a sensibilidade ao contexto na configuração e execução de missões de sensoriamento. O objetivo é manter um baixo custo de comunicação associado às soluções propostas. Esta preocupação se dá pelo fato da comunicação aumentar o consumo de energia em redes de sensores, o que é um problema importante para nós sensores com limitada fonte de energia, i.e. baterias. No caso de nós sensores móveis, esta limitação pode não ser relevante, uma vez que seu movimento deve consumir uma quantidade muito mais expressiva de energia do que a comunicação. Neste caso, o problema se relaciona à estabilidade dos enlaces, bem como ao curto intervalo de tempo disponível para transmitir e receber dados. Logo, o melhor é comunicar o menos possível.

Com relação à interação entre nós sensores estáticos, os problemas de disseminação e alocação de missões de sensoriamento são estudados e uma solução que explora o uso de informações locais é proposta e avaliada. Esta solução emprega agentes de software móveis que têm a capacidade de tomar decisões autônomas através do uso de informações de contexto local. Para redes de sensores móveis, o problema estudado se refere a como transferir missões entre os nós sensores de acordo com seu movimento e localização em relação aos locais onde as missões devem ser executadas. Para tratar este problema, uma abordagem baseada em agentes móveis é proposta, na qual os agentes implementam a migração das missões de sensoriamento usando informações de contexto geográfico para decidir a respeito de suas migrações. Para redes de sensores com sensores estáticos e móveis, a cooperação entre eles é abordada através de um mecanismo com inspiração biológica para realizar a realizar a entrega de dados emitidos pelos sensores estáticos aos sensores móveis. Para isto, explora-se uma analogia baseada no comportamento de formigas na construção e seguimento de trilhas.

As soluções propostas são flexíveis, sendo aplicáveis a diferentes domínios de aplicação. Resultados experimentais evidenciam sua escalabilidade, avaliando, por exemplo, seu custo em termos de comunicação, além de outras métricas de interesse para cada uma das soluções. Estes resultados são comparados aos atingidos por soluções de referência (solução ótima teórica e baseada em inundação), indicando sua eficiência. Estes resultados são próximos do ótimo teórico e significativamente melhores que aqueles atingidos por soluções baseadas em técnicas de inundação.

Palavras-chave: Sistemas de vigilância; Redes de sensores sem fio; Sensores cooperativos; Sensores móveis; Networking biologicamente baseado; Sensibilidade ao contexto.

1 INTRODUCTION

1.1 Preliminary Considerations

Wireless sensor networks have gained an increasing importance over the last years, due to several interesting and valuable applications that can make use of this emerging technology. It was pointed out by Business Week (GROSS, 1999) as one of the 21 most important technologies for the 21st century. The advances and miniaturization of computing, sensing and communication devices have provided means to the development of cheap, but intelligent, sensor nodes that can work in different network configurations. These networks are able to provide a variety of more or less pre-processed data to more extensive and often centralized back-office information systems with different applications (AKYILDIZ et al., 2002).

Many early applications of sensor networks appeared in the area of military systems, such as SOSUS (Sound Surveillance System), which was an array of acoustic sensors deployed on the bottom of the ocean to detect soviet submarines, installed by the US Navy during the cold war (WHITMAN, 2005). The studies performed by DARPA (Defence Advanced Research Projects Agency) sponsored a great number of military projects that evolved the sensor network technology (CHONG; KUMAR, 2003). At that time, most of the sensor networks were sets of wire connected sensor nodes. With the advances in radio communications, sensor nodes were equipped with radio transceivers, and the use of wirelessly connected sensor nodes created possibilities of sensor network employment in several new applications (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). These new possibilities called the attention of several research communities, both in academia as well as in the industry, which then applied and experimented with wireless sensor networks in a variety of application areas such as wildlife monitoring (LIU; MARTONOSI, 2003), home assisted living support (RAS; BECKER; KOCH, 2007), rescue and disaster assistance (ERMAN; HOESEL; HAVINGA, 2008), fire prevention and control (FOK; ROMAN; LU, 2005), among others.

A class of WSN based systems with employment in both military and civilian applications is area surveillance. Surveillance systems can be used for borderline or area monitoring. These usages can be applied to, for instance, homeland control, road traffic monitoring, forest fire monitoring, electrical transmission lines monitoring and security area surveillance, such as those used to provide security assurance of goods storage areas in harbours (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005) (XU, 2003). An important feature that this kind of system must provide is the flexibility to perform their missions in accordance with their specific users' requirements. This is

important due to the different needs and the dynamic nature of the environment in which those systems are deployed, in which unexpected changes may occur at any time. These changes should not interfere in the systems' performance, which must continue performing their missions. It is also important to notice that these systems can be employed to perform more than one mission simultaneously in the area where they are deployed. This requires a flexible way to setup the sensor nodes in accordance to the information that the users are interested in. At the same time, the sensor nodes have to autonomously configure themselves and form a network in order to be able to accomplish the different missions that they are requested to perform.

Depending on the application and mission needs, surveillance systems in general need to use different kinds of sensor nodes, which gather a variety of raw data that are then merged and refined. As a result of this process, higher level information about a given phenomenon is generated and finally delivered to end users (BARDELABEN, 2003). However, the ability of a sensor network to successfully perform this work depends on the cooperation among the different kinds of sensor nodes available in the network, so that they can efficiently contribute to the achievement of a common goal. This cooperation is particularly challenging when additional to the different types of data that the sensor nodes can provide, they may also differ in other capabilities, such as their mobility (AKYILDIZ; KASIMOGLU, 2004).

Mobility is an important capability that allows spatial relocation of sensor nodes (GIAMBERARDINO; GABRIELE, 2008), which can be provided by mounting them in autonomous vehicles, such as unmanned aerial or ground vehicles (UAVs or UGVs) (KIM; GU; POSTLETHWAITE, 2008) (POPA; MYSOREWALA; LEWIS, 2009). Other alternatives are mobile sensor networks composed by nodes that cooperate in a more opportunistic way, such as those composed by portable devices like cell-phones or PDAs (TEI et al., 2005), or even vehicles in urban areas (LEE et al., 2009b).

An interesting approach is the cooperative usage of both static and mobile sensor nodes (AKYILDIZ; KASIMOGLU, 2004), which is an emerging trend of particular interest to surveillance systems (ERMAN; HOESEL; HAVINGA, 2008). This promising combination allows the implementation of surveillance systems composed of simple and inexpensive static sensor nodes, which can be deployed in a large number of units, cooperating with fewer, but more sophisticated and expensive, mobile sensor nodes.

Besides the concern about the functional behaviour that has to be addressed by the necessary cooperation mentioned above, being composed by static, mobile or both types of sensor nodes, WSN have several other concerns that need to be taken into account.

Static sensor nodes are usually resource constrained in terms of processing power, available memory and energy budget. Thus, cooperation mechanisms in WSN that use such sensor nodes should be simple, implemented by low complexity algorithms requiring little space for data storage. Moreover, considering that any interaction among sensor nodes is carried out via wireless communication links and observing that communication often is the most expensive task in terms of energy consumption (MINI; LOUREIRO, 2009), the cooperation can only be considered efficient if it rationally uses communication, thus saving energy resources.

For mobile sensor nodes, generally processing power, memory and energy resources are not specifically constrained, considering that they are carried by sufficiently large vehicles. However, due to dynamicity of the network topology caused by their movements, the communication with other nodes is quite unreliable, which is a

concern that has to be considered (LEE et al., 2009b). Moreover, depending on the applications, secrecy is a great concern and unnecessary usage of wireless communication may expose the system to hostile entities, which is the case of military applications (LEE et al., 2009a).

Observing these concerns, this work aims to address the cooperation problem among wirelessly connected mobile and static sensor nodes, diminishing the need for internode communication. The proposed strategies to tackle the problem are based on bio-inspired mechanisms and mobile software agents. The proposal considers surveillance systems as the primary motivation scenario, which is used to support the problem formulation. However, it is important to highlight that the developed techniques are general enough so that they can be applied also in other application scenarios.

After presenting the motivating scenarios, the statement about the problems that are addressed in this work is done, followed by the description of the goals, scope delimitation, and the achieved contributions. Finally, this introductory part describes the methodology used and outlines the content of the entire text.

1.2 Motivation

Growing demands for new products and technological advances support one another in a closed chain fashion in which the market demands push the technology forward and the new possibilities created by new technologies give new demands (ADNER; LEVINTHAL, 2001). This “never-ending” innovation loop is not new (ADNER; LEVINTHAL, 2001); it has worked for many years in different technology driven segments and the same can be observed in the development of WSN technology and its applications (GROSS, 1999). This innovation mechanism in the WSN area brought to reality many applications in the recent years, and a number of others are expected in the near future (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

In different domains, from health care (ALEMDAR; ERSOY, 2010) to military systems (LEE et al., 2009a), the variety of WSN applications is huge. In this work the focus is concentrated on surveillance system applications of WSN, which have utility both in the military and civilian domains. Border control, wild life monitoring, disaster relief, road traffic control, law enforcement, and security are some of the possible applications, to name a few.

Surveillance systems aim to detect predefined events of interest, which may represent danger or threat, such as non-authorized vehicles or people in a given area, or catastrophic occurrences, such as fire or flooding, depending on the application (ORDOWER; DIXON; LYNCH, 2010) (XU, 2003).

A general purpose surveillance system may present specific needs that motivate the use of static, mobile or combinations of both static and mobile sensor nodes. The choice of the type of sensor for a given purpose has different reasons, such as the system secrecy, or the unsuitability of one type of sensor due to environmental constrains. Thus it is most likely that only static, only mobile or the combination of both types of sensor are desired in many surveillance systems depending on the circumstances in which these systems are used (CURTIS et al., 2010). Thus, they have to offer these different possibilities to their users.

Static sensors nodes on the ground, usually also called unattended ground sensors (UGS) (MCQUIDDY, 2010), range from simple and cheap sensor platforms such as small piezoelectric sensors, which are more commonly used in WSN (AKYILDIZ et al., 2002), up to more sophisticated and expensive image sensors, such as infra-red cameras (MCQUIDDY, 2010). Despite of their sophistication, in general such sensors have constraints on the available energy, as they are driven by batteries that have to last as long as possible, due to practical issues in replacing or recharging them, such as hostile or hazardous environments. The potential usage of such sensor nodes are greatly enhanced when they are connected with other peer sensor nodes in a network, which allow extraction of higher semantic information by the application of fusion techniques (NAKAMURA; LOUREIRO; FRERY, 2007).

Mobile sensor nodes provide an extended usage for the sensors, as they are able to change their position according to the needs of the surveillance missions. This allows the use of image sensors mounted in mobile platforms that can cover large areas, thus increasing the range of actuation of these sensors compared to the situation in which they are static (GIAMBERARDINO; GABRIELE, 2008). Being mobile on the ground or in the air, this capacity to change the sensor's position provides an important enhancement in surveillance missions, and the usage of a number of such mobile sensors in network is a natural extension of this approach (GROCHOLSKY et al., 2004), which is being massively used in military surveillance (BARDELABEN, 2003).

Combining static and mobile sensor nodes in an integrated network is a promising approach that allows the deployment of advanced surveillance systems (ERMAN; HOESEL; HAVINGA, 2008)(XIAO; ZHANG, 2009) to provide area monitoring. This combination of sensors has its motivation based on the desirable complementary features that they can provide to a surveillance system. On one hand, simple static ground sensor nodes usually employed in WSN are generally very cheap, allowing a massive deployment. However, they are only capable of providing basic data, like presence or movement detection (AKYILDIZ et al., 2002). On the other hand, typical mobile sensors are more expensive and more sophisticated, such as radars, visible light or infra-red cameras; mounted on mobile platforms such as cars (LEE et al., 2009b), robots or airplanes that can move in two (POPA; MYSOREWALA; LEWIS, 2009) or three (KIM; GU; POSTLETHWAITE, 2008) dimensions. Moreover, mobility provides the singular feature mentioned above that enables the coverage of large geographical areas (GIAMBERARDINO; GABRIELE, 2008).

Besides the specific advantages of both static and mobile sensors, they also have individual drawbacks (YICK; MUKHERJEE; GHOSAL, 2008). A WSN composed of just static simple sensors is not capable of providing information as rich as delivered by more sophisticated ones. However, a WSN composed just by sophisticated sensors, static or mobile, may not be possible to be placed or installed where needed and may also have prohibitive costs depending on how sophisticated the desired type of sensors are, and how expensive the platform that supports the required mobility is. A combination of static and mobile sensors can overcome the limitation of the data provided by the simple static sensors and the high costs of the sophisticated mobile ones [YZ09]. This is possible by using the static sensors to trigger the displacement of the

mobile ones to the areas where they are needed, which can be done by creating and sending alarms to request the mobile sensors to move to those indicated locations. With this approach, a reduced number of mobile sensors can be employed, reducing the overall system cost while keeping the ability to sense semantically rich data.

Using only static, only mobile and/or a combination of sensor nodes, the overall surveillance scenario can be described as a large area in which smaller sub-areas represent areas of interest for different surveillance system users. Different users can submit several different sensing missions that have to be performed in separate areas of interest to acquire data about a given phenomenon, thus they are defined as mission areas (MA). Figure 1.1 presents the overview of this overall scenario, in which three MAs are defined.

As can be observed in Figure 1.1, the mission areas can be of different sizes and shapes. In the figure, three examples of shapes for MAs are presented, namely a circular, a rectangular, and a squared one. Moreover, these MAs may overlap with each other. The three examples of MAs have different sensor nodes performing surveillance. MA-1 has a WSN composed only of static sensor nodes, while MA-2 has only mobile sensor nodes, and MA-3 has both types of sensors. For example, considering a system that uses UAV-carried sensors, MA-1 can be seen as a non-fly zone (NFZ), thus just static sensor nodes can be used to perform its surveillance. Another possibility is the case in which cars are used as mobile sensors and MA-1 is a region with mountains without road infrastructure. MA-2 can be a region in which the use of static sensor nodes is avoided due to, for instance, secrecy restrictions. This is a case where mobile sensor nodes are mixed with other mobile nodes and they are supposed to be undistinguishable from an observer perspective, such as the example of urban surveillance presented in Mobieyes (LEE et al., 2009b), in which a vehicular sensor network (VSN) is used to provide urban surveillance. In MA-3 both mobile and static sensor nodes are combined to perform the surveillance mission. Each of the MAs presented in Figure 1.1 represent a sub-scenario of the overall surveillance scenario, with its own restrictions and peculiarities that allow or require the usage of different types of sensor nodes, or a combination of them.

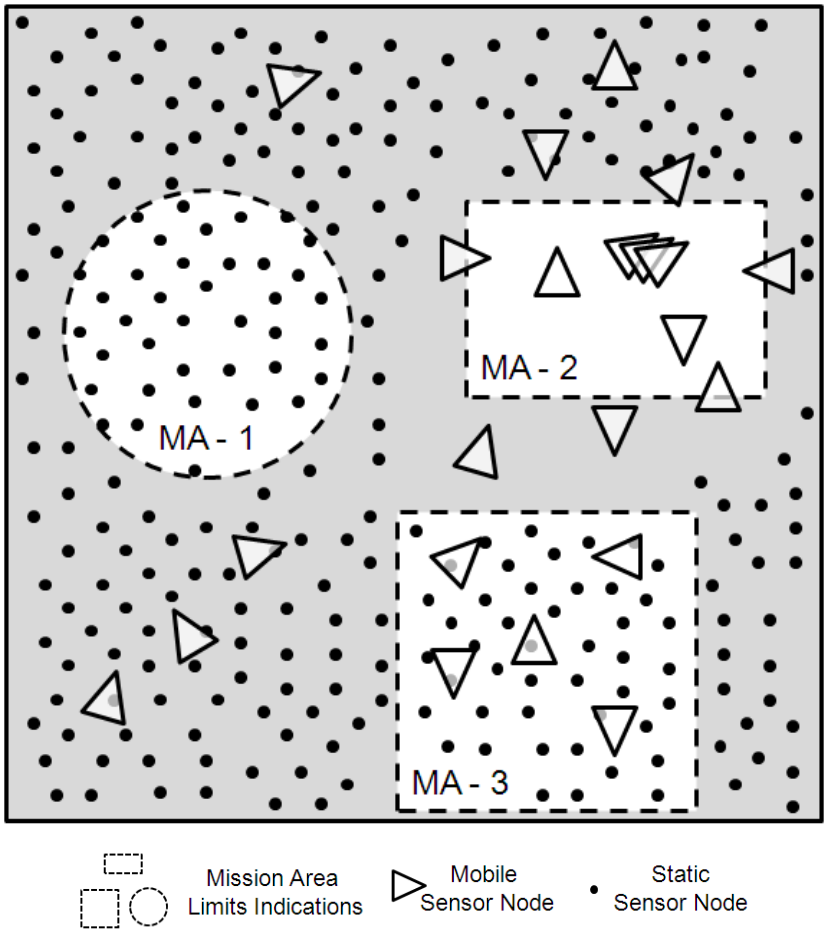


Figure 1.1: Overview of a set of surveillance application scenarios and related mission areas.

The users of such surveillance systems specify sensing missions by specifying the type of information on which they are interested. Usually high level languages are desired to specify missions, such as query-based languages (MADDEN et al., 2005), possibly even including a graphical user interface (GUI) to facilitate the system usability. Indeed, surveillance systems' usability is a big issue as the enormous amount of data generated by them require appropriate processing and presentation so that it can be made really useful (SCERRI et al., 2010). The mission specification provided by the user should minimally contain parameters such as the location where he/she wants the mission to be performed, i.e. the MA, the time bounds for the mission accomplishment and the type and amount of data that should be collected.

Additional parameters can be used to specify sensing missions, depending on the configuration possibilities offered by the system. According to these possibilities and the type of sensor nodes that compose the system, users can specify for instance: group formations for mobile sensor nodes (BEARD et al., 2006), specific movement patterns or paths to be followed (GAO 2010), data fusion (NAKAMURA; LOUREIRO;

FRERY, 2007) or aggregation (RAJAGOPALAN; VARSHNEY, 2006) directives, time constraints (BEARD et al., 2006), among others.

1.2.1 Cooperation among Static Wireless Sensor Nodes

The first scenario is connected to MA-1 in Figure 1.1 and provides an example of an ordinary case of static WSN used for area surveillance (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). There is a number of static sensor nodes spread in the MA according to some distribution which can be random or uniform following a defined pattern. The sensor nodes communicate with each other via wireless links within a tuneable, but limited, communication range. Due to the broadcast nature of wireless media, all nodes in the range of a sending node receive the sent messages.

These sensor nodes perform simple measurements, such as differences in magnetic field and CO2 concentrations, among others, which can indicate the occurrence of events of interest, such as the appearance of vehicle, people or a fire spot. A number of data fusion and aggregation techniques can be used to extract information from the raw data provided by these nodes (NAKAMURA; LOUREIRO; FRERY, 2007). As a result, high-level information can be delivered to end users representing the result of a submitted sensing mission.

Users specify the sensing missions, which are then sent to the WSN. Once these missions are specified and arrive at the WSN access point node or sink, they have to be disseminated through the network and allocated so that the sensor nodes can perform them. Depending on the access point's location in relation to the MA, the missions have to be forwarded via other sensor nodes until they arrive at their respective MA. The missions have to be informed to sufficiently many nodes within the respective MA so that the required number of sensor nodes needed to accomplish the missions get knowledge about them, and thus can be allocated according to their performance characteristics.

This whole process, from the missions' specification until the engagement of sensor nodes in their accomplishment, can be called mission setup which is depicted in the sequence of Figure 1.2.

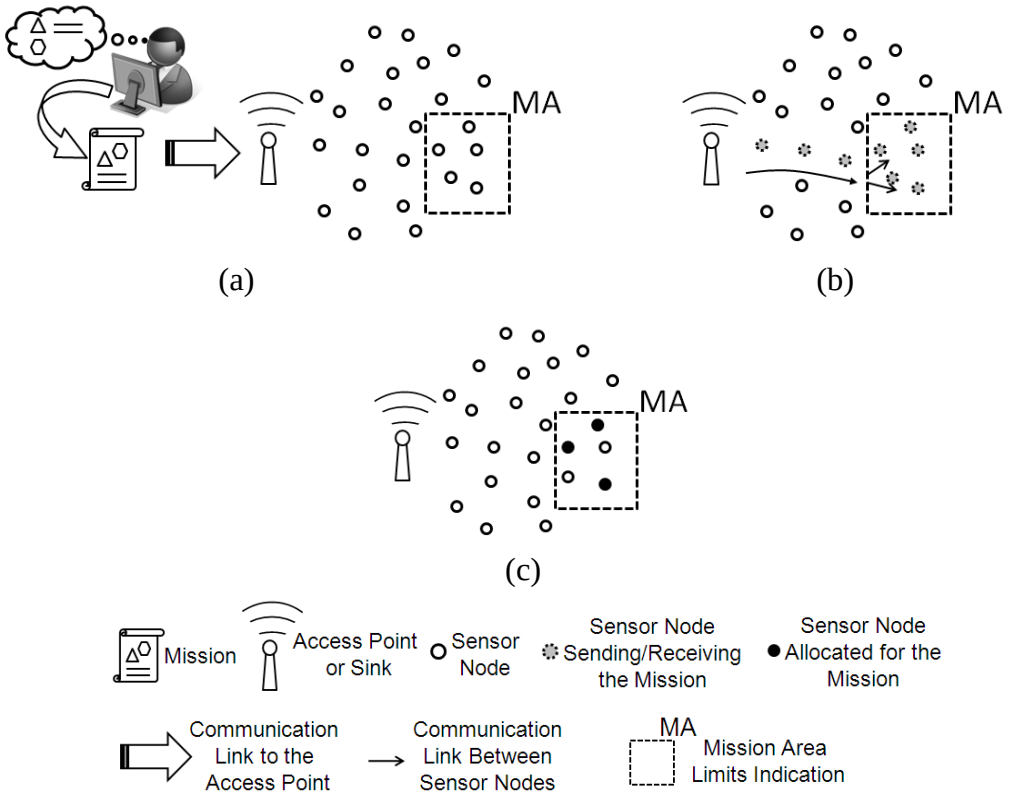


Figure 1.2: a) Mission specification; b) Mission dissemination: c) Mission allocation.

Figure 1.2a illustrates a user specifying a sensing mission, which is processed and sent to the access point. Figure 1.2b presents the mission dissemination; from the access point until it reaches the nodes in the MA informing them about the mission. Finally, in Figure 1.2c the nodes inside the MA that are allocated to perform the mission are highlighted.

1.2.2 Cooperation among Mobile Wireless Sensor Nodes

The second scenario is related to MA-2 covered exclusively by mobile sensors, thus constituting a mobile WSN (MWSN). In this scenario the sensors nodes are moving into the area as well as leaving the area. While moving inside the MA-2, they perform the respective mission. When they leave the area, they try to handover the mission to an incoming sensor node.

Considering a simplified scenario in which two sensor nodes have as primary goal the coverage of a given area, implemented by moving according to predefined movement patterns, they can perform another specific sensing mission, within a MA, concurrently while moving. As their primary goal is not this mission, they can handover the mission between each other in accordance to their movement, so that the mission is assumed by the node that is moving towards the MA. Figure 1.3 presents this situation in which two sensors scout an area in which a minor area (MA) is defined and has a mission to be performed within its borders. One of the sensor nodes, S-1, does the

scouting according to movement paths from west to east, while the other, S-2, perform a movement from north to south. Figure 1.3a shows the situation in which the S-1 is carrying and performing the mission inside the MA, which is graphically denoted by its gray colour. In Figure 1.3b, S-1 is leaving the MA and communicates with S-2, performing the handover of the mission to this sensor node. Figure 1.3c presents the situation in which S-2 holds the mission, denoted by having its symbol coloured in gray, and it is starting to perform it by entering in the MA.

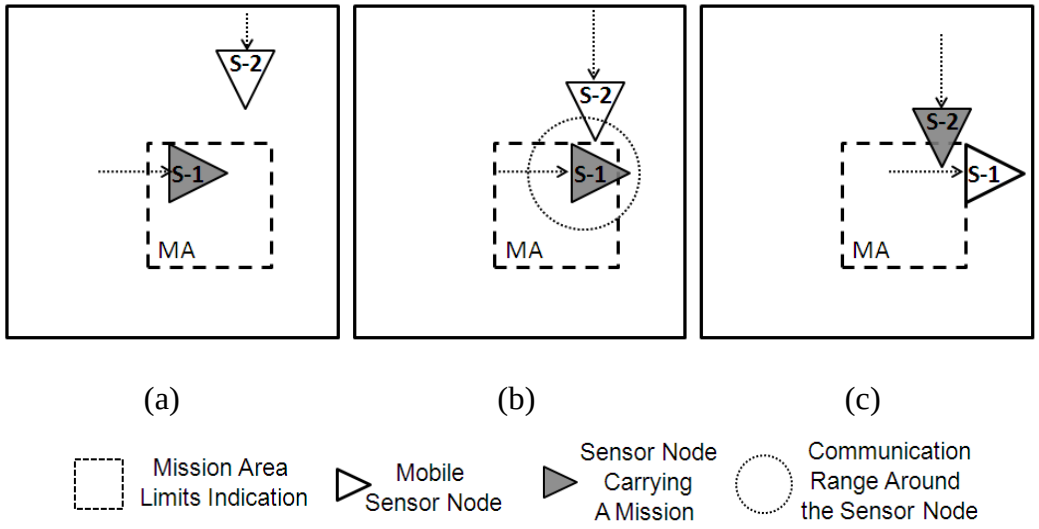


Figure 1.3: Mission handover between two mobile sensor nodes.

As the sensor nodes do not have, in principle, any agreed movement pattern or routes that would facilitate the missions' handover, the network formed by them to perform the surveillance of the area can be considered an opportunistic network (CONTI et al., 2009). In this type of network disconnections and reconnections are frequent, and the nodes opportunistically can deliver their messages when they meet each other. In this studied scenario, a sensor node leaving a MA and carrying a mission may not meet any node to which it can handover the mission. Thus it may perform this handover later when meeting another node outside the MA. This node in its turn can perform the mission itself, if it comes into the MA, or handover it to another node in the case in which it changes its direction to another location outside the MA. This handover-chain between pairs of nodes can be seen in a larger scenario as a migration of a mission from a given place, where its hosting node currently is located, towards a node inside the mission area.

The node density in the MA and in its surroundings plays an important role in how the interactions among the nodes will be made, as well as how frequent they will occur. Notice that the example presented in Figure 1.3 is just an illustration of how the mission handover would be performed. In fact the usefulness of such cooperation among mobile nodes is easier perceived in large networks with many nodes, in which the mission

migration being performed as a sequence of handover actions between several nodes can be observed.

Regarding the node density aspect, this scenario has a great potential application in contexts such as urban surveillance. In this type of environment a VSN could be used to monitor traffic congestions, levels of pollutants or collect data to prevent terrorist attacks (LEE et al., 2009b). Another possibility is a network of heterogeneous nodes such as vehicles and cell-phones could be used to monitor the levels of acoustic noise (acoustic pollution) within specific MAs. In post disaster operations, such networks of heterogeneous nodes can be used to perform sensing missions that allow data acquisition by rescuer teams that can be used to prioritize the response of emergency situations (TEI et al., 2005). Due to this large number of applications that fit in the second scenario, the urban surveillance performed by sensors carried by ground vehicles is the one selected to be further explored as part of this thesis.

1.2.3 Cooperation among Static and Mobile Wireless Sensor Nodes

The third scenario, related to the handling of MA-3, combines the usage of both static and mobile sensor nodes. Considering surveillance of large areas using WSNs composed by both mobile and static sensor nodes, the choice of mobile sensor node types is an important subject and problem area [YZ09].

Mobile platforms or carriers on the ground, such as Unmanned Ground Vehicles (UGVs), have the ability to move a sensor to a place where it is needed, but the sensor's operational area may be reduced due to either the terrain's geography or other obstacles, despite the existence of approaches that try to address these problems (KEWLANI; IAGNEMMA, 2008). Mobile platforms in the air moving in three dimensions (or just two dimensions at some predefined height), such as Unmanned Aerial Vehicles (UAVs) (QUARITSCH et al, 2010), may provide better results. These platforms have the ability to move the sensor to the desired locations from above, avoiding obstacles that they might face if they were on the ground. But even in this case where UAVs are used, a choice has to be made among different types of UAVs. Small UAVs, such as those produced by MLB (MSB, 2011), are much cheaper than large UAV platforms, such as Predator and Globalhawk (STANSBURY; VYAS; WILSON, 2009). Small UAVs make possible the usage of not just a few but many UAVs in a system, increasing system capabilities and enhancing system robustness by redundancy. Moreover, small UAV platforms enable the system to perform missions in certain sensitive regions, on which large platforms would not be able to access, such as urban environments (FREW; BROWN, 2008). Based on these arguments, the interest in small UAV platforms is rational, and they are considered in the study of this specific scenario.

This scenario considers a situation where some UAVs are flying over the MA, following a random or a predefined movement pattern. They are equipped with sophisticated sensors, such as visible light cameras, infrared cameras, and SAR/ISAR radars, while static ground sensor nodes equipped with simpler sensors are deployed on the ground within the MA limits. The UAVs' sensors provide more detailed information compared to the static sensors on the ground. The number of UAVs is however much lower than the ground sensor nodes, and the idea is to make them work cooperatively,

so that they complement each other, as previously discussed. The distribution of static ground sensor nodes is assumed to ensure that they cover the whole area sufficiently well, and, when they observe and can identify a possible event of interest, they trigger an alarm that is sent to the UAVs, e.g. asking them to perform more accurate observations with help of their more sophisticated sensors.

Similarly to the first scenario, the sensor nodes communicate with each other via wireless links within a communication range, which allows all nodes within this range to receive the messages sent by a node.

The system behaviour is defined as follows. Ground static sensor nodes are configured to detect phenomena indicating possible threats, which are defined by a set of threshold levels related to their measurement values. When the acquired measurements reach a configured threshold level, a “match” with the detecting criteria is achieved. In the occurrence of a match, the corresponding ground sensor node issues an alarm, which is received by all nodes that are within its communication range.

Alarm messages contain a timestamp, the position of the issuer node, and the type of the possible threat. The two first components of the alarm enable its unique identification, avoiding alarm duplication. This work assumes the atomicity of the events reported by the alarms, which means that each alarm that is sent indicates a different threat. Consequently, if the indicated threat is a group of persons or vehicles, they are handled as a single entity. This assumes that neighbouring nodes on the ground cooperate to aggregate decision information needed to identify and characterize threats before issuing an alarm.

The main elements of the described scenario are presented in Figure 1.4. The figure includes an illustration of the detection of a possible threat made by a ground sensor node, which is highlighted by a black filled circle to distinguish from the other sensor nodes. This node issues an alarm that is received by all its neighbour nodes in range. One of these neighbours relays the alarm, which is then received by the neighbour static sensor nodes and by a nearby UAV.

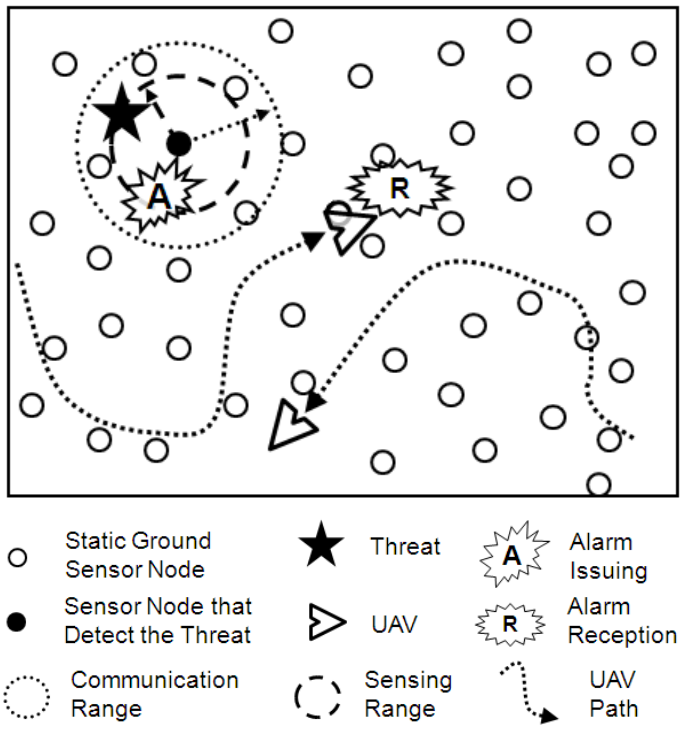


Figure 1.4: Overview of the cooperation among static and mobile sensor nodes, from the alarm issuing to its delivery.

At the occurrence of an alarm, one UAV, equipped with more sophisticated sensors, is selected and has to fly towards the area where the alarm was issued. With its more sophisticated sensors, the UAV is able to gather further information about the possible threat and then confirm or deny it as a threat, e.g. an intruder or a fire spot.

1.3 Goals and Scope Limits

Within the context of the above presented surveillance scenarios, there are a number of problems to be addressed. The goal of this work is to provide solutions with low communication costs to support cooperation among static and mobile sensor nodes aiming to accomplish surveillance missions. Taking into account the considered differences in relation to the mobility capabilities of the sensor nodes, this general goal is divided into three more specific goals:

a) For the static sensor nodes cooperation, the goal is to provide a strategy to distribute the sensing mission to the network nodes and select the more appropriate ones to perform the mission while reducing the communication among the nodes to carry out these actions, in order to save energy that is spent due internode communication;

b) For the cooperation among mobile sensor nodes, the goal is to provide a solution that helps missions to reach their respective mission areas and maximize the time that

they will stay inside these areas within the time interval in which the mission should be accomplished, observing communication and energy constraints;

c) For the cooperation among static and mobile sensor nodes, the goal is to provide a strategy to allow them to communicate and select an appropriate mobile node to respond to an alarm while minimizing communication to save energy.

From this scope are excluded the study of problems like languages and abstractions to specify sensing missions (MADDEN et al., 2005), as well as user interfaces. Sensor data fusion (NAKAMURA; LOUREIRO; FRERY, 2007) or aggregation (RAJAGOPALAN; VARSHNEY, 2006) is not considered either. The scope also excludes subjects related to sensor nodes mobility problems, such as UAVs' aerodynamics (RAUF et al., 2011), joint movement formations (BEARD et al., 2006), collision avoidance (ALEJO et al., 2009), time constrained team work (BEARD et al., 2006), team work area coverage (BEARD et al., 2006), path planning (GAO 2010) and object tracking using mobile sensors (KIM; GU; POSTLETHWAITE, 2008). Security, which is an important issue in WSN (WANG; ATTEBURY; RAMAMURTHY, 2006), is also out of the scope of this thesis work. Dependability issues are taken into account in the considered assumptions for the proposed solutions, but a deeper and exhaustive discussion about this subject is not in scope of this thesis, in spite of its importance. For instance, message delivery assurance is an important issue that is out of the scope within the dependability context.

1.4 Problem Statement

In view of the above presented scenario, the delimitation of the scope and the statement of the goals of this thesis; the problems to be addressed can be formulated.

The main problem to be addressed in this thesis can be abstracted as the classical data dissemination problem in wireless sensor networks (AKYILDIZ et al., 2002) (AKYILDIZ; KASIMOGLU, 2004) combined with a resource management problem in distributed systems (TANENBAUM; STEEN, 2007). In this problem sensing missions are seen as tasks and sensor nodes are seen as available resources needed to process these tasks.

In relation to the data dissemination, the problem is how to efficiently transmit data among the sensor nodes so that they can accomplish a given sensing mission. The task and resource allocation is per se a problem studied in the domain of Optimal Assignment Problem (OAP) (GALE, 1960), and for example used to define the Multi-Robot Task Allocation problem (MRTA), as discussed in (DRESSLER, 2007), or more specifically to sensor and actuator networks (SANETs) as discussed in (AKYILDIZ; KASIMOGLU, 2004).

Considering the motivation of this thesis work, the difference in the mobility capabilities of the sensor nodes that compose the network provides distinct instances of this combined problem, as described as follows.

Observing the first scenario presented in Section 1.2.1 from the injection of the mission into the network (Figure 1.2b), the mission dissemination has first to reach the MA and then inform the nodes within its limits. Thus a mechanism has to be provided

to take care of this part of the process. By arriving at the sensor nodes, the mission has to be allocated to nodes that can perform it, finalizing the setup. Considering the dynamicity of the operation conditions, such as topology changes due to interferences, and the energy constraints of these nodes, strategies aiming at efficient mission setup have to be flexible and economic in terms of communication to save energy. These problems are called a) sensing mission dissemination and b) sensing mission allocation, respectively.

For the second scenario, an opportunistic network of mobile sensor nodes is intended to perform sensing missions within a MA. Depending on the application, it is not mandatory that these nodes are exclusively intended to perform these missions. Taking advantage of the opportunistic network formed by them while moving, they can be “hired” to perform missions when their movement direction matches with the MA of a given mission. The mission dissemination problem in this scenario is how to transfer the missions among the sensor nodes so that they reach and remain in sensor nodes located in their respective MAs. The part of the problem related to the mission allocation is intertwined with the dissemination, as once the mission reaches a node that matches the location criterion associated with a MA, or is moving in its direction, the mission is considered allocated to the node.

In the situation described by the third scenario, the first problem to be handled is how to make the messages sent by the static nodes reach the mobile ones in an efficient way. Then, considering mobile sensors with different capabilities, the choice of a suitable one to respond a request from a static node has to be addressed. Again, the energy concern has to be considered, as even if it may not be a major problem for the mobile sensor nodes, the static ones cannot afford a huge amount of energy consumption when interacting with the mobile nodes. The former problem is called alarm delivery, as the requests sent by static nodes are considered as alarms, while the latter is called alarm handler assignment.

For all considered scenarios, it is assumed that to reduce the communication among the nodes, mechanisms that can diminish the number of messages that are sent by them have to be provided. This assumption disregards communication problems that would affect the messages’ reception, which would require retransmission schemes, affecting the number of messages that are sent and received in the network.

After this brief introduction that identifies the problems to be addressed in this work, they are discussed in details in the following sections.

1.4.1 Sensing Mission Dissemination and Allocation in Static WSN

The setup phase of WSN poses problems related to how to provide information about a sensing mission to the sensor nodes and then how to engage a group of appropriate nodes to effectively carry out the mission. Important to notice that it is assumed that missions submitted to the network are able to be handled by its sensor nodes, i.e. the network has the required resources to engage to perform the missions.

Moreover, considering dynamic environments in which changes affect the sensor nodes’ capabilities to perform sensing and networking, they have to be able to adapt

their behaviour so that they can be prepared to accept new incoming missions. Thus, the problems to be addressed are:

a) Sensing mission dissemination;

b) Sensing mission allocation.

The first problem is how to disseminate the mission among the nodes in the sensor network, i.e. how to make the sensor nodes aware about the missions in an efficient way. A trivial solution would be to broadcast mission directives to all nodes, but this is not an efficient approach, considering that not all information concerning a given mission is interesting to all nodes.

The second problem is how to split the workload among the nodes after receiving a new mission. A trivial solution for this mission allocation problem would be to take centralized decisions and send the specific part of a given mission to the specific nodes that will have to handle it. However, a central decision maker, with an oracle view of the network, must have information about all the operation, network and environment conditions in order to adequately fulfil its function. This would require an unnecessary message traffic consisting of control information sent from the entire network to the central oracle node, thus overusing communication resources, besides the effects of delays caused by such unnecessary traffic. Additionally, this central decision maker would represent a single point of failure, fact that would negatively impact the overall system reliability. On the other hand, a decentralized decision, autonomously taken by the nodes and made in a dynamic context, can give better performance within given time and communication constraints, besides increasing the system reliability.

Part of the second problem is how to adapt the nodes' behaviour to divide the workload based on the operating or network conditions. This is important because these conditions can influence the way a mission should be divided. If the system needs to wait for the operators' intervention, maybe it becomes too slow, and thus the autonomous decision making capability can give large benefits so that new incoming missions are allocated to suitable nodes. Moreover, the system has to be robust to overcome undesirable situations, such as hotspots and inefficient node distributions, which may compromise its results.

Orthogonal to the above mentioned problems is the concern about communication. As the activities performed by sensor nodes to disseminate missions and to inform other nodes about their status use communication resources, communication is tightly related to the problems discussed above and possible solution approaches. Moreover, keeping in mind that communication is the most expensive activity in terms of energy consumption (MINI; LOUREIRO, 2009), these mentioned activities impact the usage of energy resources, which is a major concern in WSN due to the limited energy resources (AKYILDIZ et al., 2002).

1.4.2 Sensing Mission Dissemination in Mobile WSN

Considering the second scenario, it is expected that there are several concurrent missions and related mission areas, like MA-2, within a larger area covered by a surveillance system. In this large area, different types of mobile sensor nodes move according to a variety of movement patterns and a number of sensing missions are

requested to be performed. Each mission is related to a specified mission area (MA) part of the large surveillance system area.

It is assumed that every mobile sensor node that populates the surveillance area has the necessary processing and sensing capabilities to perform submitted missions. Moreover, they are not supposed to be bonded to any of the missions, but they are expected to cooperate and perform any of them upon request. However, differently from the sensing mission dissemination and allocation problem described above, which considers only static sensor nodes, in the second scenario it is assumed that all sensor nodes are mobile. In this case, the dissemination problem is different and continues until a mission is finished, because the nodes do not necessarily remain inside the MA, but may move around in a larger area possibly entering and leaving the specified MA multiple times. The nodes thus have to be able to handover missions to other nodes according to their current location and destination. Moreover, as it is assumed that all sensor nodes have the capabilities needed to perform the mission, the condition that enables a node to perform a mission is thus that it is located inside the MA, hence there is no allocation problem as discussed in the previous subsection, as the dissemination by its own allocates the mission to the node that is carrying it.

From the point of view of a mission, it is important that the mission is kept within the limits of its mission area. Hence, the mission dissemination problem to be addressed concerns how to provide an efficient way in which the nodes can handover a mission from one to another so that missions hosted by nodes located outside the MA eventually will reach nodes located inside their respective MA. The reason for this behaviour is to maximize the time in which missions remain inside these MAs during the time interval on which the mission is to be carried out. The situation presented in Figure 1.3 is a particular case, in which a node that is holding a mission and is leaving the mission area, S-1, meets another node that is entering the mission area, S-2. Besides the issue about the node density mentioned in Section 1.2.2, from a broader perspective in which an area with a greater number of nodes and MAs, considering the injection of new missions into the network, the problem is how to select mobile sensor nodes in an efficient way. Examples of questions that arise in this scenario are: is it worthwhile that a node handovers a mission to any node that it meets, or should the nodes' movement direction be considered in order to minimize useless handovers?

Useless handovers are those that do not lead the missions closer towards their MAs. Such handovers can just maintain the current situation of a mission, in which it is outside its MA, or can make the situation worst, i.e. it can handover the mission to a node that will carry the mission even far away from its MA. This last case is obviously undesirable from the functional perspective of the system. The first kind of useless handover is also undesirable but due to other aspects such as: secrecy (LEE et al., 2009b), communication links instability (SOLTANI; MISRA; RADHA, 2008), and energy preservation (TEI et al., 2005) (in cases when energy resources need to be considered).

In the light of above observations, the sensing mission dissemination problem in mobile WSN can be summarized as how to efficiently perform the mission handover between nodes so that the missions eventually reach nodes inside their respective MAs,

and that the time the missions are hosted by nodes located inside these MAs is maximized, while the number of handovers is minimized.

1.4.3 Alarm Handling in Cooperative Static and Mobile WSN

The cooperation and interaction among static and mobile sensor nodes in WSN is strongly influenced by energy and communication constraints. In the third scenario, the static sensor nodes are thus considered to have severe restrictions on energy consumption, as already discussed.

Mobile sensor nodes in the form of small UAVs, as considered in the scenario, cannot carry the same load as larger UAVs, and this directly affects their communication capabilities and range. Another constraint linked with the load capacity is that small UAVs must use their energy in an efficient way, since they are neither able to carry much fuel nor large batteries. This impacts not only the communication subsystem, but also restricts the operational range of small UAVs, limiting their cooperation possibilities. Thus, even though the mobile sensor nodes do not have severe energy restrictions, as the static sensor nodes on the ground, their energy consumption must still be carefully considered.

In order to combine both types of sensor nodes and make them work cooperatively, communication is a must. However, the problem is how the static sensor nodes locate the mobile ones in order to deliver their messages. Moreover, how to select the mobile sensor node that is suitable to respond to a given alarm, considering that they may have different capabilities. Thus, these problems can be defined as follows:

a) Alarm delivery: how to efficiently deliver or route alarms from ground sensor nodes to the mobile sensors (UAVs), and;

b) Alarm handler assignment: how to decide and assign a mobile sensor (UAV) to handle a given alarm.

The first problem can be handled in at least two different ways: 1) via a central entity that collects information about all alarms and then distributes them over the mobile nodes; or 2) via a decentralized information handling and distribution process, in which alarms are directly delivered to mobile sensors via the static sensor nodes cooperating with each other to relay the alarms. Each option leads to different possible effects. However, considering that a centralized solution would hardly scale up with a larger number of static sensor nodes, mobile sensor nodes and alarms; and hence due to the high costs in terms of communication that can be expected, a decentralized alternative seems to be more reasonable (MUTAMBARA, 1998). Thus the problem (a) is how to provide a mechanism that efficiently performs the alarm delivery without a central coordinator node.

The second problem (b) relates to the decision if an available mobile sensor node is suitable or not to handle a given alarm. This matching decision should consider the characteristics of each accessible mobile sensor node and an analysis of the threat that has triggered the alarm, and what it requires in terms of sensor capabilities to be confirmed by the mobile sensor node. This second problem also affects the first problem, as it may impact the way the alarm delivery is performed.

1.5 Approaches

To face the above identified problems, different software agent techniques are used. The most important of these techniques are multi-agent systems using mobile software agents and biologically-inspired agent behaviours and algorithms. These techniques have been chosen first because multi-agent approaches provide natural models to design highly distributed system, such as WSN, with abstractions and protocols for decentralized cooperation among independent system entities (WOOLDRIDGE, 2002). Then the choice for mobile agents is based on the fact that they provide flexibility to (re)deploy software in the network, besides the fact that it provides means to combine data and intelligence (code) in communication messages (WOOLDRIDGE; JENNINGS, 2002). Biologically-inspired algorithms provide simple and naturally decentralized solutions with inherent features, such as self-organization, which are very useful in WSN (DRESSLER; AKAN, 2010). Hence, they provide means to solve the identified problems as described in the following.

1.5.1 Sensing Mission Dissemination and Allocation in Static WSN

For the problems related to the setup of static sensor nodes, needed to perform a given sensing mission, the approach proposed in this work uses a multi-agent system (WOOLDRIDGE, 2002) in which mobile software agents disseminate the sensing missions among the sensor nodes and decide about their allocation, cooperating with stationary software agents in the sensor nodes. The idea is based on that local decisions can be taken by agents in the sensor nodes, avoiding unnecessary exchange of data among them and the need for one or several network coordinators. By avoiding these messages, the overall communication in the network decreases as well as the energy consumption. To stress the importance of the trade-off between communication and local computation, it is worth to mention a rule of thumb that holds for many WSNs, which states that the transmission of 1 bit is roughly equivalent to the execution of 1000 instructions in terms of energy consumption (HILL, 2003).

However, there is a limit in the reduction of inter-node communication. This limit is the necessary communication that is required to guarantee that the nodes get information about the mission requirements from the users, i.e. what the sensor network has to provide as result of a mission. On one hand, in traditional approaches, sensor nodes are told exactly what each node should do and when, which is an approach that requires much communication between the central nodes, those nodes that distribute these tasks, and the sensor nodes (HONG et al., 2008). This comes from the fact that, in order to partition the tasks and allocate them to the most suitable nodes, the central nodes have to periodically or sporadically collect status data from the whole network. This represents a considerable overhead in terms of data traffic and thus energy consumption, in addition to other problems such as single point of failure sensitivity. On the other hand, new approaches try to make the nodes as autonomous as possible, making them able to take decisions about what they should do based only on general data requirement directives, i.e. missions' specifications. Even requiring some additional data exchange, depending on the proposed solution, this type of approach is

able to explore the above mentioned minimal inter-node communication limit, in order to reduce the total amount of network traffic due to network management, thus saving a significant amount of energy (HEIMFARTH et al., 2010a). Agent-based approaches for WSN belong to the new type of strategies that provides autonomous behaviours to sensor nodes (TYNAN; O'HARE; RUZZELLI, 2006). Thus, motivated by the need for local autonomous decisions, the adoption of a multi-agent approach to deal with mission dissemination and allocation problems is proposed.

The reasoning mechanism that empowers the software agents' decisions to perform the sensing mission allocation is based on a probabilistic decision procedure exploring local information without the need for additional communication among the nodes. Additionally, a biologically-inspired mechanism based on the behaviour of bees is also adopted to make the solution for sensing mission allocation more robust. This approach mimics the bees' behaviours when searching for food in the nature, and this analogy is explored by one type of mobile software agent part of this multi-agent approach to distribute information among the sensor nodes.

1.5.2 Sensing Mission Dissemination in Mobile WSN

As discussed in Section 1.4.2, the effectiveness of the opportunistic execution of the sensing missions by mobile sensor nodes while they move across general surveillance areas towards specified mission areas depend on how efficient and precise the missions reach and remain inside their MAs. This efficiency also depends on how good the missions can be handed over from node to node.

Sensing missions are characterized by requests for measurements of a physical phenomena that have to be done within a given time window in a certain area. Thus, they can be seen as services that run on top the platform provided by sensor nodes, using the nodes' sensing and processing resources. Moreover, the decision about if to keep a mission in a given node or instead to make a handover to another node may vary according to different criteria. In the studied scenario, this criterion is related to the mission area, but it can be anything else, and be different for different missions. Hence there is an important coupling between the mission and the control about if it should stay in a node or be handed over to another.

Mobile software agents provide a modular approach to implement services providing intelligent behaviours that allows the management of decisions based on specific criteria, which can be related to their own movement (LANGE; OSHIMA, 1999). This feature provides a perfect match between the description of the missions presented above and what the mobile agents' model can offer. Thus, this thesis work presents a multi-agent approach in which mobile software agents are used to implement the sensing mission dissemination in mobile wireless sensor networks. A mission is then encoded into and carried out by a software agent, which has the capability to reason about if it should transfer or not itself to another node as to follow the mission directives.

Despite the match between the mobile software agent model and the identified needs, an additional element is needed to the success of this approach. This element is the usage of context information to support the agents' decisions. Thus a context aware

approach is adopted to make the agents capable to take advantage of context information so that more appropriate decisions can be made, i.e. the agents decides if they should stay in their current node or migrate to another. This work investigates different approaches (represented by levels of intelligence) on how to use location information to support the agents' decisions to maximize their stay in their respective mission areas. As this is a general proposal based on the scenario presented in Section 1.2.2, it has to consider sensor nodes of different types, including the resource constrained ones. Thus, energy concerns have to be taken into account. This issue motivated the statement about the minimization of the mission handovers between sensor nodes in Section 1.4.2, which is addressed in this work.

The proposed approach aims to increase the efficiency of the agents' migrations among the mobile sensor nodes, i.e. increasing their stay within their respective mission areas and minimizing the number of agent migrations between nodes. In this solution it is not considered the collection of the data acquired by these agents, which can be addressed by solutions like those presented in (LEE et al., 2009b), in which a collector agent may perform this task.

1.5.3 Alarm Handling in Cooperative Static and Mobile WSN

The combination of both static and mobile sensor nodes represents a promising solution to enhance WSN to better support wide area surveillance applications. Despite the promising results, the joint use of these two types of sensor nodes and networks presents problems that need to be overcome in order to make the system work properly, as discussed in Section 1.4.3. The cooperation among two types of nodes presents increased complexity if compared to traditional WSN, composed by only static or only mobile nodes. This added complexity comes from various concerns that range from efficient energy usage by resource constrained sensor nodes to the efficient employment of mobile sensors during operation, by driving them to the places where they are most needed.

The proposed mechanism to address the related problems aims at providing efficient communication among sensor nodes, minimizing the number of exchanged messages in the network, thus also decreasing the overhead in terms of energy consumption. To achieve this goal, a biologically-inspired approach is formulated, which explores the concepts of artificial pheromones and stigmergy (BONABEAU; DORIGO; THERAULAZ, 1999) as a means to disseminate the information needed to make sensor nodes able to cooperate. To address the first problem described in Section 1.4.3, the goal of the artificial pheromones left by the mobile nodes is to provide information about their current location to the static sensors nodes, so that these nodes can efficiently route and deliver alarm messages to the mobile ones. This solution is further enhanced to consider mobile sensor nodes with different characteristics, which are represented by differences in the flavours of pheromones left by them to provide information to the static sensor nodes. This enhancement addresses the second problem, as it allows the selection of appropriate mobile sensor nodes to respond a given alarm.

1.6 Contributions

According to the problems and the goals addressed in this thesis work, and based on the adopted approaches, the main contributions presented are listed bellow:

a) ***Sensing Mission Dissemination and Allocation in Static WSN*** - providing support for WSN setup by means of decentralized decisions about sensing missions' dissemination and allocation by inserting mobile intelligence into the sensor nodes.

This contribution consists of the proposal and experimental evaluation of a multi-agent approach in which mobile software agents are able to carry sensing missions to the sensor network and its nodes. By using local information acquired from the sensor nodes in a given neighbourhood, the agents are able to make decisions about the allocation of nodes to perform a given mission. Hence, this proposal helps in the effort to save energy, by reducing the communication among sensor nodes. This effect is achieved firstly by avoiding conventional flooding in the sensing mission dissemination, and secondly by taking local decisions in the nodes in order to avoid additional communication among them. Experimental results indicate significant savings in the number of exchanged messages compared to conventional flooding strategy. This contribution was first published in (FREITAS et al., 2009e), (FREITAS et al., 2009b). Then deeper analysis were presented in (FREITAS et al., 2009c), (HEIMFARTH et al., 2010b) and (FREITAS et al., 2011c), which motivated the enhancement proposed in (FREITAS et al., 2010b). In (FREITAS et al., 2011d) the overall contribution is summarised.

b) ***Sensing Mission Dissemination in Mobile WSN*** - propose the deployment of sensor network services by disseminating sensing missions to suitable mobile nodes that are not exclusively dedicated to perform sensing missions, but that can also be used for this purpose.

This part of the work investigates an approach based on the use of mobile software agents to implement sensing missions that also can migrate among sensor nodes according to the nodes' geographical locations. The idea is to deploy a "virtual sensor network", which is implemented by the software agents that run on the mobile nodes. The goal is to keep the agents hosted by the nodes that are located in the areas of interest for the specific mission that is performed by the agents. This proposal includes the usage of geographical context aware decision mechanisms to support this virtual sensor network, exploring, testing, and comparing different strategies. This contribution was first presented in (FREITAS et al., 2010d) and further enhanced and analysed in (FREITAS et al., 2011a).

c) ***Alarm Delivery and Alarm Handler Assignment*** - propose and investigate means to allow the cooperative usage of mobile and static sensor nodes to perform area surveillance missions.

The major challenge in the effective cooperation between static and mobile nodes is mainly related on how to provide information to static nodes about current location of the most suitable mobile sensor node. This work provides a contribution in relation to

this topic, by presenting an approach to the routing and delivery of messages from static to mobile sensor nodes. This approach makes use of biologically-inspired concepts and is implemented by means of artificial pheromones that first help to find and then indicate the direction of the movement of the mobile sensor nodes. This is used to route alarm messages addressed to the mobile nodes. Moreover, the approach is further enhanced to provide the selection of a better fit between the type of the mobile sensor node and the type of the event that should be handled. A number of experiments are performed to explore different scenarios and characteristics of the proposed solution, highlighting several relevant aspects that are discussed. This contribution was first presented in (FREITAS et al., 2009a) and (FREITAS et al., 2009b), having its first experimental results published in (FREITAS et al., 2009d). A more detailed description of the proposal and the acquired results is presented in (FREITAS et al., 2010a), while more experiments and analysis are presented in (FREITAS et al., 2010c) and (FREITAS et al., 2010e). An overall summary of the contribution is presented in (FREITAS et al., 2011e).

Crosscutting the above listed main contributions is the concern about energy consumption. All the listed contributions provide efforts aimed to minimize the communication among the nodes that compose the WSN. Thus, these main contributions have to be understood in the context of efforts made to reduce the energy consumption.

1.7 Methodology

A set of modelled scenarios are used as experimental base to study the defined problems related to WSN based area surveillance. The detailed specification of these scenarios sets the scope and limits for the experiments. Small scale experiments are used to test selected approaches to get a better understanding of their suitability to tackle the problems. The results of these experiments support the design of larger scale experiments to validate the proposed solutions.

The methodology to conduct these experiments and the analysis of the obtained results follows the guidelines for randomized experiment designs described in (BOX; HUNTER; HUNTER, 2005). These guidelines explain how the variables should be controlled to construct block designs so that it is possible to observe variations of interest in randomized experiments as well as how to analyse and interpret the results. Randomized experiments are used due to the large number of possibilities that the scenarios under concern provide, for instance, areas with different dimensions, different possible number of nodes, different nodes' placement in the areas, and movement patterns. Hence, the use of this methodology makes it possible to achieve reasonable case coverage as well as representative and unbiased results, which can be used to draw conclusions about the general behaviour of the applied solutions in the considered scenarios.

A comparison between reference solutions and the achieved results is the method adopted to objectively evaluate the quality of the proposed solutions. A subjective comparison with related works is also performed. This comparison must be subjective

since none of these related works provides experimental results that are directly comparable to the achieved ones.

Simulation is the main method used to experimental validation. The motivations for the choice of computer simulations are twofold. The first is that it is much cheaper to build a large scale simulation, with hundreds or even thousand of nodes, compared to a physical prototype. The second one is related to the fact that specific hardware and software used in demonstrators often consumes significantly more development time and effort to be correctly configured in order to be employed in evaluations performed to assess the behaviours of the network running the proposed solutions; which could hinder the progress of the work, and possibly also lead to a loss of focus. Additionally, the use of analytical models does not provide a good alternative either, due to the high complexity of the studied scenarios. Analytical models for these scenarios would be too simplified to be tractable, which would considerably diminish their capability to express many relevant aspects that are able to be handled in simulations.

In order to choose an appropriate simulation tool, the peculiarities of the type of sensor networks that is focused on in this work (composed by sensor nodes with possible different movement capabilities), were studied and compared with the features provided by some available simulation tools (SINGH; VYAS; TIWARI, 2008) (LESSMANN et al. 2008). The comparison indicated that none of the analysed simulation tools fitted perfectly to the kind of networks intended to be studied, neither the adopted solution approaches. The main considered points were: facilities to provide simulations of wireless networks; presence of mobility models; usability and user-friendly extension mechanisms; visualisation and result reporting support; and open-source availability enabling software modifications if needed. Taking into account all these criteria, a simulator developed especially for wireless networks, and successfully used also for wireless sensor networks, was chosen. This simulation tool is ShoX (LESSMANN; HEIMFARTH; JANACIK, 2008) originally developed at Paderborn University. By extending this simulator with additional features required to perform the intended simulations, such as the possibility to run simulations in which different nodes have different movement patterns, or different communication ranges, a new simulator was created, called GrubiX (HEIMFARTH; FREITAS, 2011).

Additionally to simulations, a small scale demonstrator prototype was also developed implementing a selected algorithm presented in this thesis work. The presentation of this demonstrator has not the same goal of the simulation experiments, but only to assess the feasibility of the proposed approach using COTS software and hardware.

1.8 Thesis Outline

In addition to this introductory chapter, this dissertation consists of nine chapters. Chapter 2 provides background information about wireless sensor networks, in which an overview of the main problems in this research area and a summarized description of traditional approaches to handle them are presented. Moreover, as mobile software agents and biologically-inspired approaches are used in the solutions proposed by this

work, Chapter 2 also includes a brief description of basic concepts of the techniques used by these two approaches.

Chapter 3 describes the proposed solution for static WSN setup. An overview of the proposed solution for the problems stated in Section 1.4.1 is presented, followed by a definition of a sensing mission and the considered assumptions. Then, the details of each part of the proposed solution are described. A presentation of the results and a related discussion conclude the chapter.

Chapter 4 describes the mobile agents solution for the mission dissemination among mobile sensor nodes, to address the problems described in Section 1.4.2. Different levels of intelligence to decide about the agents' migrations are explored in the provided solution, which have their results compared and discussed.

Chapter 5 provides the proposed solution for cooperative use of static and mobile sensor nodes. The first part of this chapter brings important definitions and assumptions, which is followed by the description of the pheromone-based solution to deliver alarms sent from the static to the mobile sensor nodes. Enhancements in the features of proposed solution are presented in an incremental fashion. Then a feasibility analysis is presented, in which the approach is analytically tested to assure its feasibility considering realistic conditions. Finally, the chapter is concluded by the presentation and discussion of the achieved experimental results.

Chapter 6 presents details about the GrubiX simulator and how the simulations used to test the proposed solutions are implemented in this tool.

Chapter 7 presents the developed demonstrator prototype that assesses the feasibility of the proposed approach by deploying one of the proposed algorithms in a physical WSN.

Chapter 8 briefly discuss some relevant aspects about dependability in WSN, which can compromise the solution approaches proposed in this work. As dependability aspects of WSN are not included in the goals of this thesis, the intention is just to provide an overview of such aspects, stating that we are aware about them, even though they are not handled in this work.

Chapter 9 discusses related works to this thesis, highlighting some similarities and differences that our work presents in relation to these other research projects.

Chapter 10 provides a concise summary of the thesis bringing the conclusions for each contribution. Finally, a discussion about ideas for future research directions based on the results obtained in this thesis is presented.

2 THEORETICAL BACKGROUND

This chapter provides an overview of important topics to understand the work reported in this thesis. These topics are: wireless sensor networks, software agents and biologically-inspired approaches in computer science. This overview does not exhaustively describe these topics, but it aims to briefly present the main concepts and ideas concerning them, highlighting those that have a relation to the content of the work in this thesis, to ease the understanding of its contribution.

2.1 Wireless Sensor Networks

Wireless sensor networks (WSNs) are distributed systems composed of a set of wirelessly connected nodes, equipped with one or more types of sensors, used to observe a phenomenon of interest (AKYILDIZ et al., 2002). Sensor nodes are then devices that encapsulate sensing, processing and communication capabilities. Studies about WSN usually consider small sensor nodes, such as the Berkeley Mica Motes (HILL; CULLER, 2002), which are tiny sensor nodes largely used in practical WSN experiments. Figure 2.1 shows the boards of the Mica2 dot Mote near a two Euro coin for size comparisons purposes.

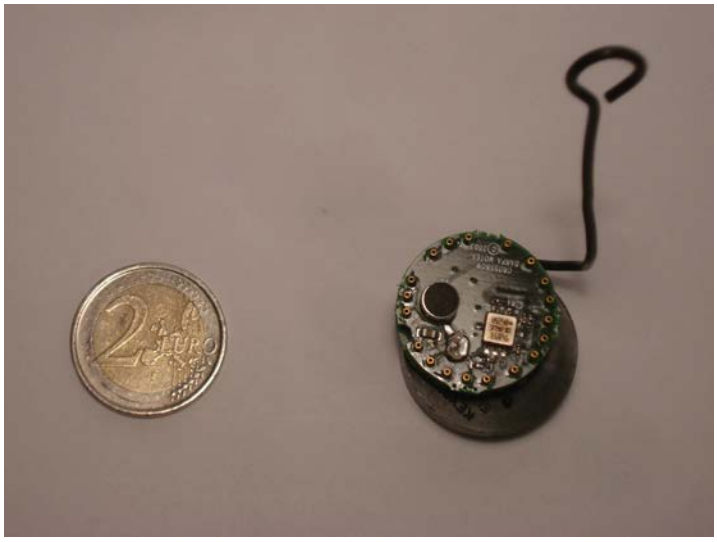


Figure 2.1: Mica2 Dot Mote.

Sensor nodes platforms vary much in relation to the hardware configurations. Table 2.1 presents selected characteristics of sensor node platforms that are commonly used to deploy demonstrators.

Table 2.1: Selected characteristics of commonly used sensor node platforms available in the market.

	SunSPOT (SUN, 2010)	Mica2/Mica2Dot (HILL; CULLER, 2002)	iMote 2 (INTEL, 2011)
CPU	32-bit 180 MHz ARM920T, 512 KB RAM	8MHz/4MHz Atmel Atmega 128L, 4KB RAM	[13–416] MHz Intel PXA271 XScale®, 256 KB RAM
Radio transceiver	ChipCon CC2420	ChipCon CC1000	ChipCon CC2420
Current (active) with radio on	104 mA	15/13 mA	44 mA (13MHz)
Current (active) with radio off	80 mA	8/8 mA	31 mA (13 MHz)
Current (sleep) with radio off	24 mA	0.01mA	0.387 mA
Battery	Lithium battery with 720 mAh	2 AA/coin cell	3 AAA
Available sensors	Accelerometer Light intensity Temperature	Acceleration/seismic, Acoustic Barometric, Magnetic Pressure Temperature	Accelerometer Humidity Light Temperature

By the figures presented in Table 2.1, it is possible to observe the impact in the energy consumption when the sensor nodes are active. Taking into account the statement about the importance in the consumption due to the usage of the radio to transmit and receive data, as mentioned in Section 1.5.1 (HILL, 2003), it is important to carefully interpret this statement. This because indeed communication is expensive in terms of energy consumption, thus it is important to reduce communication, but depending on the algorithms implemented by the application running in the WSN, the cost in terms of processing and data acquisition may also imply in high energy costs.

The sensor nodes can be static or mobile, depending on their intended usage. If the sensor nodes in a network are mobile, this network is referred as a mobile wireless sensor network (MWSN) (MUNIR et al., 2007). If a combination of mobile and static sensor nodes composes the network, the term hybrid is often used in the literature to determine this type of WSN (COLTIN; VELOSO, 2010) (REN; MA; CHEN, 2006). If the sensor nodes have different sensing capabilities, the network is usually called multi-

modal wireless sensor network (BOONMA; SUZUKI, 2007). The term heterogeneous is also used to refer to WSN composed of nodes with different capabilities, but in general this term has a broader sense, referring to WSN in which the sensor nodes may alternatively differ in other capabilities, such as computational power, communication links or energy resources (YARVIS et al., 2005). However, although being used in this context, these terms are not very consistently used in the literature, thus the authors usually define what they exactly mean when using these terms.

WSNs are accessed via special nodes called sinks or access points, which can be static or mobile. Depending on the WSN application and the multiplicity of the requesting users, multiple sinks may be present in the network. The purpose of the sink nodes is to provide an interface between the WSN and another type of network from which the end-user will be able to access the data acquired by the sensor nodes (AKYILDIZ et al., 2002). This interface may reach applications in a local area network limited to an institution, or even the internet (BOTTS, 2002).

Regarding the above introductory characteristics of WSN, their setup and operation require methods that are quite different from those used for conventional computer networks. In computer networks, the user is interested mainly in the computation performed by a given computer, whereas in a WSN the user is interested in the acquired data regardless the sensor that provided it. The integration of the sensor nodes with the physical world is also an essential concern. This requires the network to be setup and adapted according to the surrounding static and dynamic environmental conditions, as well as the user needs (ZHAO; GUIBAS, 2004). This setup influences the network operation, but also the way the sensor nodes are deployed. Static sensor nodes can be spread or be dynamically engaged over an area of interest according a given pattern to monitor a certain phenomenon. Mobile sensors can use different movement patterns to cover an area, such as linear, random-walk or circular movements (CORTES et al., 2004). Usually WSNs are very robust against failures of individual sensor nodes, due to inherent redundancy provided by deployments that use a great number of sensor nodes that serve as backup for each other in their vicinity. This aspect is especially true for static sensor networks composed of low cost sensor nodes, which allows the deployment of a large number of these nodes (SOUZA; VOGT; BEIGL, 2007).

Sensing tasks, ideally defined at a high-level of abstraction, obtain answers by a combination of individual contributions from several limited sensors that compose the whole system. This operation mode aims to make the sensing tasks more efficient by reducing the volume of data transmitted in the network, i.e. data aggregation (RAJAGOPALAN; VARSHNEY, 2006), provide information with higher quality, i.e. data fusion (WALD, 1999), or both (NAKAMURA; LOUREIRO; FRERY, 2007). Middleware solutions for WSN aim to provide these high-level abstractions, providing different levels of data aggregation or fusion (HENRICKSEN; ROBINSON, 2006).

Besides the particular way of operation, WSNs also have certain important characteristics that must be addressed in their design. The first main characteristic is related to constraints that limit the sensor nodes. WSN nodes are often small embedded devices that have limited energy supply and restricted processing performance, memory space, and communication bandwidth and range (AKYILDIZ et al., 2002). These

constraints requires that the algorithms that are executed on the sensor nodes be simple, and do not require the storage of large amounts of data. Moreover, the communication with other nodes has to be used only when strictly necessary, as additional to bandwidth restriction, energy constraint is a main concern, as the communication is an expensive task from the energy consumption perspective (MINI; LOUREIRO, 2009).

The high degree of dynamicity is another important characteristic of WSNs. Failures, non-intentional displacement of nodes, unexpected mobility of tracked events, and environment obstacles can interfere with and disturb the functionality of the network. As a consequence, network topology changes may occur, which can lead even to network partitioning, possibly isolating nodes or group of them from the rest of the network (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). However there are cases in which dynamicity can provide advantages to the network, such as controlled node mobility (CORTES et al., 2004).

Heterogeneity is also an issue that must be taken into account. As mentioned above, sensor nodes can be of various types, presenting different features, such as processing power, memory capacity, available energy, sensing capabilities, mobility, and communication range among others. Furthermore, most nodes in WSNs are small and resource constrained, thus the need for harmonization in the interactions and cooperation among different types of nodes is a must (AKYILDIZ et al., 2002).

Regarding the characteristics and operation modes of WSNs, some software design strategies are proposed. A common characteristic is the use of distributed algorithms to implement and achieve specified global sensing mission goals (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). Another desired characteristic is related to the possibility to implement adaptive applications with algorithms that achieve an efficient use of resources.

In ordinary computer networks, nodes' selection is done by using a unique identifier, for example by a node's network address; this method is successfully used because the communication in these networks focuses in transferring data between specific nodes (TANENBAUM, 2003). In WSNs, due to the overlapping deployment of sensor nodes, which results in an inherent redundancy, the user is usually not interested in acquiring the requested information from a specific node, but from a desired geographical region or from nodes which provide a given type of information. The most important issue is the information or data and not which node provides it. This is called data-centric communication, in contrast to conventional address-based communication (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

The data-centric characteristic of WSNs generates requirements for different routing protocols in which nodes are not selected by an identification address, but by the data that they provide. In fact, the node itself is not addressed, the importance is focused on the data that is being requested or monitored. As an example, in a sensor network that is used to monitor the temperature in a building, the queries are not addressed to nodes "X" and "Y", but to a location with given properties, such as "the conference room in the third floor" or "the place where the temperature is greater than a threshold" (MADDEN et al., 2005). In the same way as spatial location or position is an interesting query factor, the time of detection is also of interest. Sensor characteristics may also be

of interest, such as accuracy or stability of the acquired data. As an example, requests can be sent exclusively to sensor nodes that provide a given desired measurement accuracy.

Information retrieval is also atypical, as data from several nodes may be combined to fulfil application requirements. The so called “virtual nodes” address this issue, in which specific patterns of data routing through the network are defined according to the application requirements, defined as queries’ (BONNET; GEHRKE; SESHADRI, 2001). This process can provide meaningful information by means of applying data aggregation or fusion mechanisms over the flow of raw data gathered by several nodes, providing time and/or space correlations (MADDEN et al., 2005).

Since sensor nodes are usually deeply embedded in the physical world, their interaction with the environment plays an important role in this kind of network. Several changes in the environment may occur, such as weather conditions changes, movement of obstacles, besides intrinsic network events, such as node failures. The network must be able to cope with these classes of problems in order to keep its usefulness (ZHAO; GUIBAS, 2004).

Still related to their surrounding environment, it is observed that WSNs are often deployed in environments that are harsh or difficult to access, such as battle fields or remote areas in which the modification of a configuration or the replacement of a software component must be achieved remotely without direct physical access. This is the case for instance in the ZebraNet project (JUANG et al., 2002). ZebraNet is a project that has as goal to monitor Zebras in their natural habitat so that the behaviour of these animals can be studied. Thus, when the sensor network was designed, an important requirement was that after its deployed, any modification in the nodes software should be performed remotely. The reason for this requirement is the fact that if a physical access to the sensor nodes was needed, these possibly frequent disturbances in the animals’ routine could interfere with their normal behaviour and invalidate the collected research data.

System lifetime is another essential issue in WSNs, due to the fact that batteries cannot be easily replaced in many of the application scenarios (AKYILDIZ et al., 2002). The definition of system lifetime varies widely, and there is no firm consensus in the literature. However, some understandings are recurrent in the literature and can be regarded as acceptable definitions for the end of a WSN lifetime, such as the time of the first failure; the moment when the network is disconnected and turned off for good; or the moment from which the network cannot provide anymore its services due to some reason (SOUZA; VOGT; BEIGL, 2007). More sophisticated definitions exist, approaching concepts that add more strict requirements to the WSNs, such as the one that defines the WSN lifetime as the time interval during which the network can provide the quality of service (QoS) requested by the user (CHEN; VARSHNEY, 2004).

If compared to other embedded computer networks, such as industrial ones, WSNs have some key distinct characteristics. The wireless communication is one of these characteristics, which imposes several problems that are not present or are easier to handle in wired networks. For instance, the control of real-time requirements in wireless networks is much harder than it is in wired ones (KUNERT, 2010). Moreover, taking an

example from industrial networks, the client-server architecture, usual in such networks, does not fit the characteristics of WSN, which has an intrinsic decentralized nature (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

2.1.1 Challenges in WSN Research

The development of software to carry out WSN based missions present challenges which mainly comes from the particular characteristics of WSN, as those presented above. WSN applications impose specific requirements that make WSNs design, implementation and deployment particularly challenging tasks (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). Military surveillance applications, for instance, provide a number of examples of such challenges. Secrecy, endurance, accuracy, flexibility are some of these requirements, to name few of them (BARDELABEN, 2003).

By their inherent goals, WSNs for military applications cannot be exposed to the hostile forces, otherwise they will not accomplish with their mission either because they will be either destroyed or fooled by the enemies. Once deployed, a military surveillance WSN is desired to last as long as the information that they provide is needed. Thus concerns about energy savings and remote software management, similar to what was discussed to the ZebraNet mentioned earlier in this chapter, have high importance. Accuracy is another important aspect, as inaccurate measurements may lead to erroneous decisions, or even confusion and misinterpretation of the actual situation. Flexibility in the military application context can be reported to software management for example, as depending on the current goals of the performed sensing missions, the possibility to deploy new data fusion or aggregation mechanisms is highly desirable.

Similarly to what happens in the development of WSN for military applications, the issue about specific domain and application related requirements is also a concern for other types of applications. This fact per se creates a challenge for system developers, which is the ad hoc nature of the WSN development (AKYILDIZ et al., 2002). This specificity usually hinders the reuse of previous developed solutions, thus increasing development costs and time. To overcome such problem, a need for appropriate programming abstractions and frameworks that are generic, but also address specificities of different WSN domains/applications, is identified. Moreover, such solutions have to be modular, which also affects the maintainability of WSN systems (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

WSNs can be composed of a variety of sensor nodes, which may vary in terms of their capabilities, related to sensing, computing, communication and mobility. System design has to deal with such existing heterogeneity among sensor nodes. Solutions for this type of problem require, for instance, data fusion or aggregation mechanisms to merge different types of measurements provided by nodes with different sensing capabilities (HEINZELMAN et al., 2004). With regards to nodes' mobility, routing mechanisms are needed, which address the presence of mobile nodes in the network (AKYILDIZ; KASIMOGLU, 2004). More generally speaking, abstractions are needed to decouple the programming and configuration of sensor nodes, in spite of how heterogeneous they are.

System scale is also a key concern in the development of WSN. The size of the network can range from tens to thousands of nodes, depending on the target application. This asks for solutions that can address such significant up scaling, while keeping the system performance and efficiency high. Another dimension of the scalability is the number of users (AKYILDIZ et al., 2002). A WSN can serve just a single user or a number of them. The number of users can increase dramatically for instance if the WSN is accessible via internet (CHRISTIN et al., 2009).

Interoperability is also an important concern related to the usability of the WSNs. A WSN can be directly accessed via a base station connected to a sink node, or via a gateway that provides a connection to a local area network (LAN) or even to the internet. Depending on how deep this interoperability and integration of the WSN with other networks is, the scope is enlarged evoking the concept of internet of things, which defines a smooth integration of different nodes, including sensors, in a huge internet-alike network (CHRISTIN et al., 2009).

Considering the dynamicity of the environment where they operate, WSNs require capabilities to quickly adapt to environment changes in accordance to what happens around the sensor nodes. Moreover, the internal state of sensor nodes has also to be followed, so that state changes can trigger necessary adaptations. To efficiently manage sensor nodes' operation, context awareness aiming to address adaptability and autonomous local decision capabilities are required. Efficiency in this context is referred to both time and energy consumption. The former is explained by the fact that if the context information has to be sent to a base station to generate decisions, the delay imposed to the sensor nodes to receive these decisions may compromise the system performance, besides the scalability problems related to this centralized approach. The latter relates to the problem of the large amount of messages that may be necessary to perform this communication between the sensor nodes and the central base station (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

WSNs use wireless communication to transmit data among the nodes. As already mentioned, considering energy resource constrained sensor nodes, communication is energy expensive (MINI; LOUREIRO, 2009). Thus, any interaction among the sensor nodes and sinks has to be designed taking into account this concern, thus aiming at protocols and algorithms that require a low number of message exchanges.

Security is another big issue in WSN. Due to the broadcasting nature of the radio media used to interconnect the sensor nodes, WSNs are particularly vulnerable to attacks that may compromise information confidentiality, integrity, and authenticity. For instance, communication among sensor nodes can be eavesdropped disclosing its content to unauthorized entities; jamming may disturb legitimate communication avoiding that messages reach their destinies; and malicious nodes may inject misleading data into the network. Solutions for these problems are particularly challenging for WSN, as considering their computing resources constraints, sensor nodes do not have, for instance, the necessary processing power to execute high demanding cryptography algorithms (ZHOU; FANG; ZHANG, 2008).

Summarizing the discussion about the challenges in WSN development, the following list provides an outline of these main concerns:

- Abstraction: programming abstractions are required to decouple from and hide the underlying sensor nodes' platforms (hardware and software), providing an easier way to (re)configure it;
- Programming: programming paradigms for sensor networks are different from traditional ones, and a need for higher level languages and methods is recognized, in order to make it easier to program the many different applications;
- Modularity: a clear distinction among the functionalities as well as among the components that provide them, so that these components can be easier replaced and reused;
- Application: despite the generality of the solutions for WSN, application knowledge is required so that the specific needs of the final applications are fulfilled;
- Data Aggregation and Fusion: raw data collected by different sensor nodes have to be merged to diminish data traffic and synthesized to provide a high-level and easily understandable format or report;
- Scalability: solutions for WSNs have to scale up both in number of nodes and users;
- Interoperability: besides the interoperability that has to be provided among the sensor nodes within the network, the concern about the access from external networks to the sensor network has also to be taken into account;
- Resource Constraints: regarding resource constraint of the sensor nodes, any solution for WSN has to be lightweight;
- Networking: networking mechanisms to support WSN operation have to consider the data-centric nature of this kind of network;
- Topology: a key concern is related to the dynamicity of the topology in a sensor network, due to a number of reasons, such as node failures or movement;
- Adaptability: adaptable and flexible behaviour is needed in order to provide adequate support under operation environment changes, performance adaptation, users' requirements changes, among other;
- Context Awareness: to be able to adapt the network behaviour, to face different operational scenarios, sensor nodes have to keep track of the changes that may require adaptations. Thus, context-awareness is a key feature that has to be supported.
- Autonomy: autonomous decision-making mechanisms placed in each sensor node are required to spread intelligence over the network, thus providing these nodes with a certain degree of independence in relation to central nodes, needed to implement a more responsive and energy efficient system.
- Security: security issues are a key concern due to the wireless communication, especially regarding applications in domains such as health care systems and military.

2.1.2 Traditional Approaches in WSN

A significant trend identified in the literature is the use of middleware based solutions to address problems in developing WSN systems (HENRICKSEN; ROBINSON, 2006). There are many motivations for this trend, such as the modularity that middleware solutions provide in contrast to monolithic software support, besides properties such as interoperability, abstraction, among others. How successful they are in providing these features and in addressing the problems listed above depends on the models that they implement (YU; KRISHNAMACHARI; PRASANNA, 2005). These models present different sources of inspiration, in which two approaches can represent the class of more traditional solutions to develop WSN systems, inspired in models for conventional computer networks, namely database and event-based (HENRICKSEN; ROBINSON, 2006).

2.1.2.1 Database-Inspired Solutions

The database model of sensor networks has become popular, providing an easy and intuitive way to retrieve information from sensor nodes. There are several proposals that follow this approach, in which TinyDB middleware for WSN is outstanding (MADDEN et al., 2005). The great success of TinyDB can be partially attributed to its innovative usage of the database model, but also to the "easy to use" application programming features that it provides. Besides TinyDB, other similar approaches are COUGAR (BONNET; GEHRKE; SESHADRI, 2001) and SINA (SHEN; SRISATHAPORNPHAT; JAIKAEO, 2001).

The idea is to provide a database like abstraction of the sensor network. Taking this point of view, a query processing system translates a high-level query to low-level commands to the sensor nodes to retrieve the desired information. This idea frees the user from having to write code for the sensor nodes in low level languages, such as nesC (GAY, 2003).

TinyDB and COUGAR are designed for use in relatively simple data collection applications, only supporting simple in-network selection and aggregation functions based on arithmetic operations and simple search criteria. SINA has similar concept, but it supports more complex queries, that allow for instance data retrieval by mobile sinks. All these three approaches use a SQL-like query language with support to temporal and data streaming.

TinyDB is more sophisticated than COUGAR in terms of energy saving enabled by calculating the frequency of the sampling to answer queries and also by the use of a routing structure that helps the nodes to route in a energy-efficiently way. COUGAR uses a schema of leader nodes to aggregate data on the way back with data that respond queries. SINA uses a hierarchical clustering mechanism that group nodes in clusters according their energy levels and proximity, in an attempt to reduce energy consumption.

A key limitation in these middleware techniques is the assumption that sensor nodes are largely homogenous. The data types/relations that will be used at every node must be agreed in advance. This is acceptable in a small size sensor network; however it represents a great limitation for networks with larger number of nodes. An important

limitation of the sensor database approach is that they are not prepared to support more sophisticated sensor types, such as cameras with image processing. Despite the support provided by SINA to mobile sinks, mobility is weakly supported in these approaches, being tightly coupled to specific queries and movement patterns, and not oriented towards a more flexible and collaborative use of sensors with different mobility capabilities.

2.1.2.2 Event-Based Solutions

Event-based approaches are based on the idea of event handling, which is suitable for many WSN in which the sensor nodes stay in a passive state waiting for the occurrence of an event, e.g. sensor reading or incoming data from other nodes, to perform processing or send data to another node. The publish/subscribe paradigm is also used by these approaches, allowing decoupling of event producers and subscribers, in which nodes interested in a certain types of data need to subscribe to them and wait for their occurrence.

Impala (LIU; MARTONOSI, 2003) is a design with a modular structure, which allows the update of certain parts of the system, without stopping the ongoing running applications. This feature provides capabilities for adaptation. It uses an event-based programming model which provides four event handlers: timer handler; packet handler; send done handler, and data handler. These four handlers allow the manipulation of data and the programming of asynchronous or periodic behaviours. Designed to support the ZebraNet project (JUANG et al., 2002), which aims to monitor the behaviour of zebras by attaching sensor nodes on the animals, it handles disconnections caused by nodes' mobility.

Mires (SOUTO et al., 2004) provides a publish/subscribe solution designed to run on top of TinyOS (LEVIS et al., 2004). In its architecture, sensors advertise the type of data that they can provide, while applications are able to select among these data, those in which they are interested in. Sensors publish the data to applications according to subscriptions. Data aggregation is also supported by subscriptions among the sensor nodes, so that nodes closer to the application user merge data from different sources before the delivery.

The main drawbacks in these approaches are the overhead to sensor network setup, presented for instance in Mires, in which the data advertisement requires that all sensors announce the data that they can provide. This initial overhead is acceptable if the nodes' conditions were supposed to remain the same during all the system lifetime, which is not a realistic condition in many applications. Thus, if a new announcement has to be done every time a change in the network or in the nodes' state occur, such as topology changes or nodes' energy level drops, the overhead tends to increase dramatically. Despite the benefits of the event handling mechanisms and the modularity provided by Impala, it does not address adaptability in an autonomous way. Rather, it depends on a central node that has to send modules with updates to change the behaviour of the sensor nodes. This means that there is no support for autonomous local decisions.

Summarizing, both database-inspired and event-based approaches present an important drawback which is the lack of autonomous mechanisms that spread

intelligence through the network nodes. This lack of nodes' autonomy creates a significant dependence to central nodes or base stations.

2.2 Software Agents

Software agent is a research topic of great interest motivated by its wide range of applications. It is difficult to precisely define a software agent, as there is an ongoing discussion in the literature about what a software agent really is, and how it differs from an ordinary computer program. An interesting viewpoint about this topic is presented in (FRANKLIN; GRAESSER, 1997), in which the authors discuss about differences and similarities among a number of definitions for software agents, and provide a taxonomy that tries to include them all. Despite this great effort, the discussion is still alive and thus other taxonomies can be found for example in (HECTOR, 2005) and (SAKARKAR; SHELKE, 2009) with the goal to provide an all-inclusive classification scheme for software agents.

As discussed in (FRANKLIN; GRAESSER, 1997), the definition of software agents really differs among several researchers on the subject. However, some common concepts are mostly present, at least partially, in these different definitions. Based on these concepts, it is possible to state that a software agent is a software abstraction or executable entity which presents the following properties:

a) Persistence: the code of a software agent runs as a continuous task and thus not only execute an action or sequence of actions and then halts as an ordinary program;
b) Autonomy: software agents are able to take decisions about what they should do next, including to stop running, without intervention of any external entity;
c) Social ability: software agents are able to interact, coordinate and possibly cooperate with other agents (e.g., HW or SW artefacts or even humans), via some kind of agent communication language;
d) Reactivity: software agents are able to perceive and respond accordingly to changes in their environment.

An additional property that is worthy to mention at this stage is pro-activeness. This property establishes that the software agents are not only capable to respond to environment changes, as defined by the reactivity property above, but they are also able to exhibit a goal-oriented behaviour, thus taking the initiative. This property is part of a largely accepted definition provided in (WOOLDRIDGE; JENNINGS, 1995), which also states that behaviour is a key concept in the agent's definition, as attributes and methods are for classes in the object-oriented approach (BRUCE, 2002). However, the pro-activity was not included in the list above because there are classifications of software agents that make a clear distinction between pure reactive, pure pro-active and hybrid types of software agents, such as the one presented in (HECTOR, 2005).

Observing the persistence property, it is possible to state that once a software agent is launched, it keeps executing continually. Hence, instead of performing a given action and then terminate by default, a software agent remains running until it decides, by

itself, to stop. An agent can also receive a request to stop. This stop condition provides a link to the autonomy property, which is very important to distinguish software agents from ordinary programs (FRANKLIN; GRAESSER, 1997) due to the fact that the possession of autonomy enables a software agent to control itself and its own acts. This property is so important that software agents and autonomous agents, as well as simply agent, are terms that are used interchangeably in the literature (FRANKLIN; GRAESSER, 1997). Related to these terms is the term intelligent agent (or rational agent), which emphasizes the use of artificial intelligence techniques, such as reasoning or learning, to steer the agent (RUSSELL; NORVIG, 2003).

The social ability focus on the communication related aspects of software agents, enabling the interaction among them. In the definition of this property presented above, two important concepts can be distinguished: coordination and cooperation.

Coordination is the ability to manage the interdependencies of activities between agents to achieve a goal (MALONE; CROWSTON, 1990), while cooperation is the process in which an agent voluntarily interacts with another towards a goal (WOOLDRIDGE, 2002), or as presented in (WANG; TIANFIELD; JIANG, 2003): cooperation is the process in which agents act together with a common purpose. A deep discussion about these concepts is provided in (WOOLDRIDGE, 2002), in which a coordination/cooperation loop is explored deriving further concepts from the possible interactions between these two.

As mentioned above, following the wide range of software agent definitions found in the literature, there is also a wide range of proposed taxonomies to classify software agents, in which some provide very detailed classifications, including implementation alternatives, such as (GEORGAKARAKOU; ECONOMIDES, 2009). Being consistent with the motivation used to provide the above definition, the taxonomy presented in the sequence tries to factor out the most important and common existing classifications. The selected classifications are:

a) Pro-activeness: defines how an agent interacts with the environment.
   a.1) Reactive: the simplest form of agent, denominated simplex (RUSSELL; NORVIG, 2003), which directly reacts to stimuli from the environment by mapping this input directly to an action using a condition ***if – then(-else)***;
   a.2) Deliberative: diametrically opposed to the reactive ones, deliberative agents use sophisticated planning techniques from artificial intelligence to achieve their goals. This agent has a symbolic model of its surrounding environment, which is used to reason about what to do next. The “Belief, Desire, Intention” (BDI) model (BRATMAN, 1987) is the most accepted technique for this goal-oriented agent behaviour (RAO; GEORGEFF, 1995);
   a.3) Hybrid: an agent that combines both reactive and deliberative capabilities.

b) Adaptiveness: is related to the ability of an agent to modify its behaviour over time.
   b.1) Adaptive: an agent is said to be adaptive if it is capable to modify its behaviour over time to adapt to its environment. Learning methods are

commonly used as technique to provide adaptive ability to agents (KOZIEROK; MAES, 1993);

b.2) Non-adaptive: agents that are not able to modify their behaviour over time are said to be non-adaptive.

b.3) Constraint-based: agents that have the ability to adapt but are restricted by a given constraint or condition are said to be constraint-based. This kind of agent is useful in critical systems, such as avionics systems, in which it is important to predict how the system will behave after adaptations (LACEY; HEXMOOR , 2003).

c) Mobility: the mobility defines the existent relation between the agent and the computer where it is being executed (LANGE; OSHIMA, 1999).

c.1) Mobile: mobile agents are those capable of transporting their execution to other computers/nodes in a network;

c.2) Static: static agents are not capable to move their execution to other computers.

d) Communication: defines the capability of an agent to exchange messages with other agents.

d.1) Communicative: communicative agents are able to send and receive messages to interact with other agents. To perform this messages exchange, communicative agents use some kind of language, such as the FIPA ACL (Agent Communication Language) (FIPA, 2002a);

d.2) Non-communicative: non-communicative do not interact with other agents via exchange of messages. However, they are capable to perform other kinds of interactions without direct communication, such as in swarm systems, in which a society of agents following simple rules manages to achieve an emerging global behaviour (PARPINELLI; LOPES, 2011).

e) Disposition: defines the attitude of an agent towards being cooperative or not with other agents (WOOLDRIDGE, 1997).

e.1) Benevolent: benevolent agents are expected to cooperate with other agents, thus sharing common goals;

e.2) Self-interested: self-interested agents act only in their own interest, thus cooperating with others only if there will be some profit in doing so. A special kind of such an agent is a competitive agent, which rivals with other agents in the achievement of a given goal;

e.3) Malevolent: malevolent agents are those that not only avoid to cooperate or even compete with other agents in the system, as the self-interested ones, but only disturb the system creating a negative impact on it, such as worms of virus.

Figure 2 summarizes the taxonomy proposed in this thesis, according to the bibliography review discussed above.

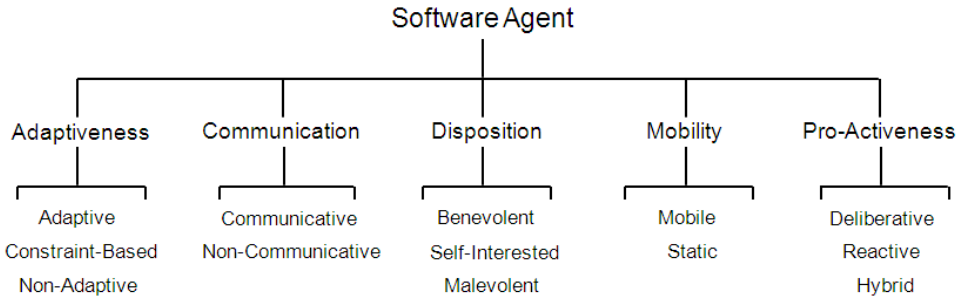


Figure 2.2: Software Agent Taxonomy.

2.2.1 Mobile Software Agents

Mobility is an important property for software agents aimed to be used in WSN applications, as emphasized by the taxonomy presented in (ORHAN et al., 2011), and has a particular interest to the work presented in this thesis.

A mobile software agent is a software entity capable of moving from one hosting node in the network to another, as defined previously. It starts its execution in one node and may continue or finish it in any other node. As an essential part, such an agent has code that defines its behaviour and also data that may be carried during its movement from one node to another. The data is called the “state” of the agent, and depending on if the agent carries such data or not during its movement, it is called stateful or stateless (CHESS; GROSOF; HARRISON, 1995).

Considering that the network nodes can also be mobile, data exchanged by them can be delivered by a message or data ferrying mechanism, which explores the store-carry-and-forward paradigm (ZHAO; AMMAR; ZEGURA, 2004). This paradigm defines that a node takes incoming data, stores it in memory and then after the node physically changes its position, i.e. it moves, it will forward the previously acquired data to another node. As agents are being communicated among the nodes, it is possible to state that this is a form of agent ferrying, which is a concept explored in (TEI et al., 2005). Agent ferrying uses the same concept as data ferrying, but instead of just data an agent is communicated. The difference between a pure agent movement and ferrying is that in the former the nodes involved in the communication that transfer the agent are static, while, in the latter, the nodes that relay the agent physically move before forwarding the agent to other nodes. Figure 2.3 presents examples of both agent movement and ferrying, in which node number 1 is the source and node number 3 is the destination. In Figure 2.3b, nodes 1 and 3 could also be mobile, which would still represent an example of agent ferrying.

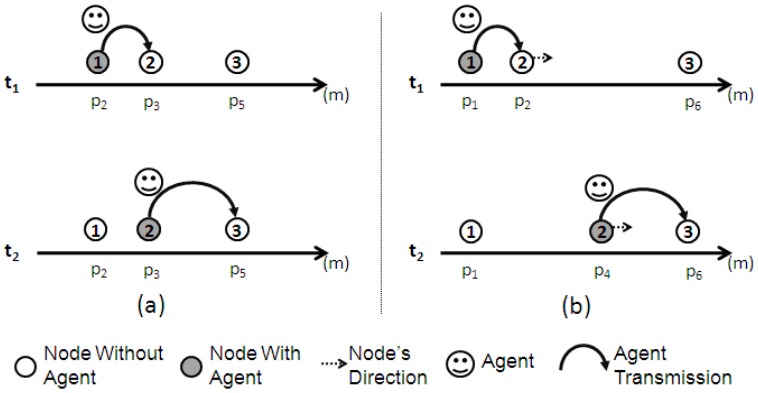


Figure 2.3: (a) Agent Movement and (b) Agent Ferrying.

Concerning the agent mobility, two more concepts have to be considered. The first is agent migration (CHESS; GROSOF; HARRISON, 1995) and the second is agent cloning (SHEHORY et al., 1998). Both use the same agent moving schemes as described above. However, in the former, the agent itself is transmitted to another node while in the latter an agent creates a copy of itself, i.e. a clone, and this clone is sent to another node.

2.2.2 Multi-Agent Systems

The discussion about social ability, coordination and cooperation turns the attention to a system, or a "society", of agents able to interact among each other. From this idea of a system of interacting agents, the concept of multi-agent systems (MAS) is conceived (WOOLDRIDGE, 2002). Usually the agents in MAS are considered to be software agents. However, in multi-agent systems the term agent does not obligatory refers only to software agents, but the agents can also be humans, robots, teams (of humans or robots), or a combination of them (KAMINKA, 2004) (SCHURR et al., 2005).

Multi-agents systems are useful to solve complex problems, in which an individual agent or program would hardly solve or is not able to solve it at all (WOOLDRIDGE, 2002). The concept of multi-agent systems is closely related to distributed artificial intelligence (DAI), which is a branch of artificial intelligence dedicated to the study of distributed solutions of complex problems using artificial intelligence (VLASSIS, 2007).

Some fundamental characteristics distinguish MAS from single agents and determine how they work (VLASSIS, 2007). The agents in a MAS may have the same or different designs. The former type is called homogeneous while the latter is called heterogeneous. These agents have to deal with an environment, which can be static or dynamic. Usually it is said that the environment of MAS is inherently dynamic from the point of view of each agent, due to the presence of the other agents. This environment is considered to be partially observable for each agent, as no agent is assumed to have the capability to access a global view of the entire environment. Moreover, this statement can be extended to the system as a whole, as the agents may not have knowledge about

the actions of the other agents, hence they do not have information about all parts of the system. This capability to observe the environment and the system may vary among the agents, thus it is possible that some agents have more knowledge than others.

The agents of MAS have no centralized control. As an autonomous entity, each such agent governs its own actions by taking its own decisions. Depending on if the agents share the same goals, they cooperate or not with the other agents (WOOLDRIDGE, 2002).

Considering the decentralized operation of MAS, an interesting characteristic that some MAS manifest is a degree of self-organization to enable the emergence of a given global pattern or behaviour by the sum of the individual contributions of each agent. The global result can present high degrees of complexity, regardless of how simple the strategies implemented by each agent can be (GABBAI et al., 2005). Swarm intelligence is a form of self-organization by collective intelligence that presents particular interest for the development of MAS due to the simplicity of the individual behaviours and due to its numerous applications (BONABEAU; DORIGO; THERAULAZ, 1999).

In the performance of self-organizing processes, agents may exchange knowledge via a standardized language, such as ACL FIPA (FIPA, 2002a), which can enhance the achieved results by increasing the knowledge base available to each agent. By exchanging knowledge, agents are also able, for instance, to coordinate their actions by managing the interdependencies among them (MALONE; CROWSTON, 1994). However, exchange of knowledge during system runtime is not mandatory and coordination can be achieved by making the agents follow a set of rules or observing pre-shared knowledge (GERVASI; PRENCIPE, 2004). The same holds for the cooperation among agents, i.e. when the agents share the same goals, in which communication can enhance the results, especially considering that the environment is partially observable from an agent's perspective (XUAN; LESSER; ZILBERSTEIN, 2001). However, again the communication is not mandatory (FLAKE, 2000).

The design of MAS that implement emergent global behaviours by using self-organization approaches is strongly influenced by systems and societies in the nature, such as models of chaotic systems and animal societies (FLAKE, 2000). Swarming intelligence is an important concept in this context, as mentioned above, and it is an inspiring new approach for WSN (ÇELIK; ZENGIN; TUNCEL, 2010), in which it in combination with other biologically-inspired mechanisms, such as genetic algorithms, are recently being proposed (CAPUTO et al., 2010).

2.3 Biologically-Inspired Approaches

Biologically inspired (or simply bio-inspired) approaches in computer science can be divided into three main areas (DRESSLER; AKAN, 2010):

a) Bio-inspired computing, which represents the algorithms aiming efficient computing, e.g. optimization algorithms;
b) Bio-inspired systems, which represent architectural solutions for distributed and collaborative system, e.g. distributed sensing; and

c) Bio-inspired networking, which represents strategies for efficient and scalable networking, e.g. massively distributed autonomic systems.

The two first areas are well established while the last one represents a relatively new research area. Bio-inspired computing is a branch of ***natural computing*** that observes how biological systems work in the nature to inspire the conception of solutions for complex computational and engineering problems (CASTRO; ZUBEN, 2005). Similarly, bio-inspired systems consider the organization of biological systems as source of inspiration to distributed computer systems architectures and bio-inspired networking to development of networking mechanisms and protocols. One of the motivations for the interest in bio-inspired approaches is the self-organizing property that many biological systems present (ASHBY, 1962), which is particularly useful for massively distributed systems, such as WSNs.

Originally, the idea of bio-inspired computing was conceived with two main purposes, in which the first aimed at modelling biological mechanisms in an attempt to better understand their functioning. To achieve this goal, these mechanisms had to be artificially reproduced as accurately as possible, so that the study of the acquired artificial model could be analysed providing the desired understanding of the studied natural system. The second purpose was the above mentioned inspiration for the development of computing algorithms and systems to solve complex problems (CASTRO; ZUBEN, 2005). In this case, the motivation is to provide alternative solutions for problems that would have high computing costs, or even be intractable, if conventional computing techniques such as linear or dynamic programming (CORMEN et al., 2001) were used.

Unlike the accurate models created to computationally study the biological mechanisms that are related to the origin of the bio-inspired computing, the bio-inspired algorithms (in the three above mentioned areas) created to solve complex problems have no ambition to perfectly model the inspiring biological counterparts. In fact the natural mechanisms are usually observed in relation to a specific property that researchers believe can be used to solve a given problem of their interest. Then, the part of the mechanism that presents such a property is analysed and modelled to provide an algorithm that express this property in the application domain of the problem under concern. It may happen that at the end of the algorithm design the similarity to the natural mechanism is very subtle, and the resulting algorithm or system even barely resembles its biological inspiration (CASTRO; ZUBEN, 2005).

The research area referred to as bio-inspired approaches is very broad (DRESSLER; AKAN, 2010). A number of biological systems provide ideas to the development of algorithms and entire systems that mimic the behaviours of natural ones. From this wide range of approaches, some have remarkable importance for being used in a number of different application domains. A non-exhaustive list of such approaches is presented as follows:

a) Artificial Immune Systems explore the properties presented by the immune system of vertebrate animals, especially learning and memory, used to solve problems that need adaptive and self-organizing behaviours (CASTRO; TIMMIS, 2002);

b) Artificial Life mimics the properties of the growing and development mechanisms of living organisms used as inspiration to develop scalable algorithms (NORTH; MACAL, 2009);
c) Artificial Neural Networks mimics the functioning of the brain, in which computing nodes of the network present the properties of neurons. They are used in applications that required learning methods, such as pattern recognition or classification (ANTHONY; BARTLETT, 2009);
d) Evolutionary Systems mimics the mechanism of growth and evolution of populations used in applications that benefit from guided random search to adapt and find a suitable solution (JONG, 2006);
e) Lindenmayer Systems model the growth structure of plants, which are used to model grammars and have several application in computer graphics (PRUSINKIEWICZ; HANAN, 1989);
f) Swarm Intelligence explores emerging collective social behaviours to inspire the creation of alternative optimizer algorithms based on social interactions of simple individuals that leads to the emergence of complex behaviour or patterns (BONABEAU; DORIGO; THERAULAZ, 1999);

Observing the context of this thesis work, which is in the WSN research area, one of the listed approaches presents particular interest, namely swarm intelligence. The support for this argument is the fact that swarm intelligence can be characterized as an approach based on self-organization, which is a remarkably important feature for the development of bio-inspired ad hoc networking solutions (DRESSLER; AKAN, 2010) and hence to WSN (DRESSLER, 2007). Artificial immune systems also present such a self-organizing feature that is explored in several WSN solutions. An example of this is an approach to redeploy mobile sensor nodes using an algorithm based on the affinity concept presented by the immune network algorithm (KUANG; CAI, 2010). Another example is the usage of negative selection to recognize anomalous patterns to implement an intrusion detection system to identify attacks to a WSN (LIU; YU, 2008). However, in artificial immune systems this feature is less evident compared to swarm intelligence approaches for WSNs.

Noteworthy that despite the interest for self-organizing approaches inspired by swarm intelligence, it does not mean that other approaches, even those not listed in this small set of samples, are not used as inspiring metaphors to solutions for WSNs. For instance, there are a number of approaches for WSN based on evolutionary systems, particularly inspired by genetic algorithms (KLEINSCHMIDT, 2009). In (ZHANG et al., 2010), a genetic algorithm is used to provide nodes' locations, in which first inaccurate information about the current nodes' locations are refined by a genetic algorithm.

Swarm intelligence is inspired on the collective behaviour of animals, such as insect colonies (ants and bees are the most commonly referred), fish schools, bird flocks or herds (BONABEAU; DORIGO; THERAULAZ, 1999). The idea of systems that implement swarm intelligent algorithms is rather simple; they explore local patterns and interactions among agents with simple individual behaviours, which execute simple tasks to build up a global sophisticated behaviour. The local interactions follow simple

rules that conduct the self-organization of these agents. This self-organization leads to the emerging of a collective intelligence called swarm intelligence.

The self-organization in swarms is achieved by the four mechanisms (BONABEAU; DORIGO; THERAULAZ, 1999):

a) ***Positive feedback*** is a behavioural rule that leads to the formation of useful structures, such as reinforcement in pheromones strength left by ants defining their trails;
b) ***Negative feedback*** is a measure that counterbalances the positive feedback in order to avoid system distortions, such as the evaporation of pheromones left by ants;
c) ***Fluctuation*** provides a random task switch among swarm individuals, which enhance the swarm behaviour by adding a creativity and innovation factor that enables the discovery of new solutions;
d) ***Multiple interactions*** among the swarm individuals are useful to spread information, such the bees' dances to show the path towards food.

Insect colonies inspire solutions for a number of problems. These solutions explore stigmergy (BONABEAU; DORIGO; THERAULAZ, 1999), a concept which is related to the positive feedback mentioned above. Stigmergy defines an indirect coordination mechanism that uses environment cues to orchestrate the actions performed by the agents. The pheromone left by ants to form their trails is a concrete example of the application of this concept. When ants find food, they lay down pheromones on the way back towards their nest. When ants coming from the nest to search food in the environment, they meet trails of pheromones and just follow those trails indicating the shortest path towards the food source previously found. The trails left by the ants form a collective memory shared by the individuals of the nest, which leads to the global behaviour driving them to the food sources by the sum of simple individual behaviours of laying down pheromone traces in the environment. This ant-foraging principle is also known as Ant Colony Optimization (ACO), which is described in (BONABEAU; DORIGO; THERAULAZ, 1999) (DORIGO; DI, 1999). Metaphors to behaviours of other insects, such as bees, are also used in a number proposals addressing routing in mobile ad hoc networks, task partitioning and allocation, clustering and multi robots controls to name few (KARABOGA; AKAY, 2009).

AntNet (CARO; DORIGO, 1998) is a routing algorithm based on ACO, which disseminate messages in the network to collect information about links and updates the routing tables of the communicating nodes accordingly. An extension of AntNet to ad hoc networks is presented in (CARO; DUCATELLE; GAMBARDELLA, 2004).

A number of solutions for routing in WSN are based on insect colony metaphor (DRESSLER; AKAN, 2010) (ÇELIK; ZENGIN; TUNCEL, 2010). In (MITTAL et al., 2010) an approach that models the data communication among nodes as ants' movements is proposed. The idea is to evaluate the accumulated pheromone level (left by the forwarded communicated messages) in the neighbour nodes before a node selects one of its neighbours to proceed with the message forwarding. This process aims to balance the energy consumption of the sensor nodes due to the message forwarding.

Flocking, schooling and herding provide concepts extensively explored in robotics, in which flight formations of UAVs is an emerging research field (LABONTE, 2009) (GURFIL; KIVELEVITCH, 2007). Approaches in this area have shown that good performance in the coordination of UAV teams, requiring little communication among team members, is achieved to collectively perform searching missions in unknown environments. Collision avoidance solutions also benefit from these concepts as discussed in (OLFATI-SABER, 2006), which presents the usage of flocking algorithms to control complex manoeuvres performed by UAV teams. WSNs also apply this type of approach, as the one presented in (ANTONIOU et al., 2009) that describes a self-adaptable congestion control mechanism for WSNs. In this mechanism, messages form flocks that flow towards the global attractor represented by the sink node avoiding regions of congestion, which represent obstacles for the flock.

2.4 Summary

By analysing the characteristics of WSNs, it is possible to notice that a main concern in this research area is how to provide suitable decentralized solutions that provide adaptability and flexibility to WSNs. These two desired features are strongly related to autonomy and self-organization. This observation motivates the study of solutions that can provide these features, such as agents and bio-inspired approaches, which is our goal in presenting their main concepts in this chapter. Moreover, the proposed solutions presented in Chapters 3 to 5 are inspired by these concepts.

3 SENSING MISSION DISSEMINATION AND ALLOCATION IN STATIC WSN

3.1 Introduction

Based on the scenario described in Section 1.2.1, the overall idea of the proposed approach to tackle the identified problems described in Section 1.4.1 is to push the responsibility for decisions about sensing missions' allocation to the network, by means of exploring local information while the missions are being disseminated. To achieve this goal, instead of having the sensing mission being passively communicated in packets describing sensing directives transmitted among the nodes, they are carried by active software agents who have the capability to take decisions while being communicated among the nodes.

The purpose of the agent-oriented approach is to avoid the need for a centralized decision maker, a node which would be the only responsible for partitioning and distributing the missions among the other nodes. To do this, such a central node would need to collect information about the entire network with certain regularity in order to have an overview of the nodes' conditions. Instead, defined missions represented by agents can enter the network via a sink node from any location, move themselves around the network until they reach the desired area where the missions should be performed, and then spread themselves among the nodes that may take part of the mission. Then, the agents in each node autonomously decide about which nodes will perform the missions, without the need for extra negotiations. The basis for this decentralized decision procedure is the information contained in the missions that the agents represent plus the information that they can get from the nodes while moving through the network. A mission provides information about the requested types of sensors to perform it, specifying their capabilities and the necessary amount of sensors that should be engaged. A mission also contains information about the location where it has to be performed and the criterion to be used in the decision about the selection of nodes (the mission allocation itself).

Figure 3.1 presents an overview of the elements involved in the proposed approach, in which it is possible to observe the representation of the mission being injected into the network from a sink node by an agent. The network is composed by nodes of different types, in which the agent will disseminate the mission in a specific area of interest, the Mission Area (MA), and it will select a subset of nodes available in this area according to their types and conditions and the mission requirements.

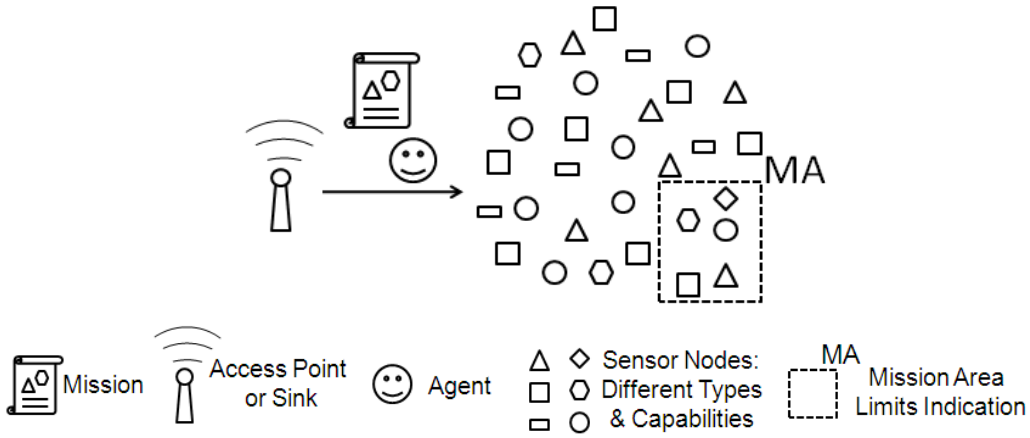


Figure 3.1: Elements involved in the proposed solution.

In summary, the proposed solution addresses the mission dissemination and allocation problem in two steps. The first one is accomplished by the injection of an agent responsible for injecting a mission into the network which is geocasted towards the location where the mission should be performed. When the agent arrives at this location, it spreads clones of itself among the sensor nodes in that location while trying to communicate as little as possible. The second part consists of the decision procedure carried out by the clones of this agent in each node that may take part in the mission accomplishment, which results in the decision if the node should participate or not. This autonomous decision made by the agents in each node represents a decentralized mechanism enabling a decentralized solution for the mission allocation in the network as a whole.

3.1.1 Sensing Mission

The concept of sensing mission, or just mission, entails all the activities that the sensor nodes must perform in order to deliver the information services based on the requests from the final user. High level dedicated languages are desired by final users to specify sensing missions, as discussed in the literature (YU; KRISHNAMACHARI; PRASANNA, 2005). TinyDB (MADDEN et al., 2005) for instance uses SQL-like input queries to provide this sensing mission specification interface to the end users. Then the requirements of the mission are extracted from this specification and sent to the network. As highlighted in Section 1.3, the focus of the contribution presented in this chapter has to be considered from this step, in which the mission is represented by an agent loaded with parameters and algorithms sent to the network. Thus, it is important to first define the structure needed to represent a mission.

A sensing mission to be carried by an agent has a structure as presented in Figure 3.2. The first field is a mission identifier, a number that uniquely identifies each submitted mission. The following field carries information about the area where the mission has to be executed, i.e. the Mission Area (MA), which is defined by its boundary points. The third field defines which sensor nodes' types and capabilities are needed to perform a given mission. The fourth field presents the evaluation criterion, which is a function of selected parameters, such as for example remaining energy levels

of the nodes and the sensors' eligibility to the mission needs. This function g defines how good the sensor node is to perform the mission, called the goodness of a sensor node for that mission. The fifth field informs the desired amount of sensors that should take part in the mission, which can be an absolute number or a percentage of the nodes that are able (eligible) to perform it. The mission may also carry additional parameters (dashed part of the mission structure in Figure 3.2), such as timing parameters, which are defined in the sixth field, that can specify periods or frequencies in which the measurements should be performed, delays, jitter and duration. The seventh field represents the procedures to handle the acquired data, such as aggregation or fusion algorithms. The eighth field describes possible actions that a sensor node may perform, such as issuing an alarm to trigger other sensors, or even open or close valves in industrial applications of wireless sensor and actuator networks (WSAN) (DRESSLER, 2007). Other additional parameters can be defined according to the needs of a specific mission.

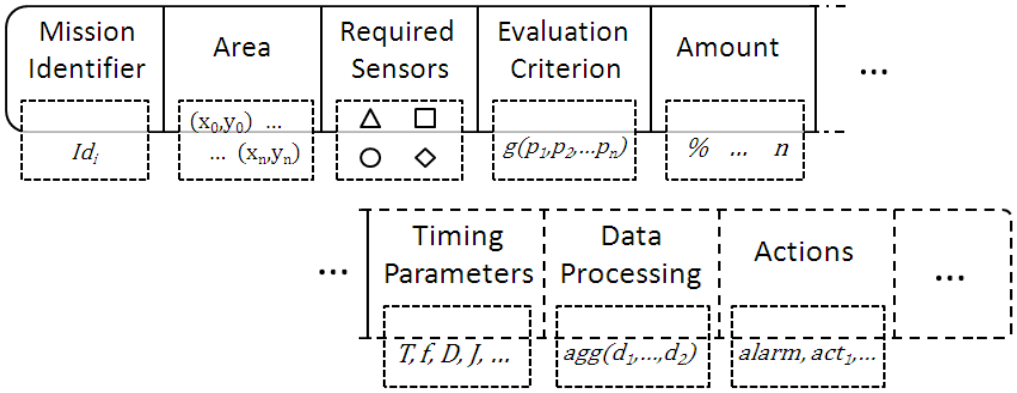


Figure 3.2: Mission structure.

3.1.2 Agent Classes

The multi-agent approach proposed to address the sensing mission dissemination and allocation problems uses three different types of software agents to support a resource aware sensor network setup. They are used for different purposes and have different properties in which the mobility is the main one, and, due to this, it is used to classify them in two different types: static and mobile. Figure 3.3 presents a class diagram, in which the bottom layer represents the concrete classes of agents used in this approach while in the upper layers are the abstract classes that categorize the concrete ones using established agent terminology (see Chapter 2), completing the agent ontology needed in this context.

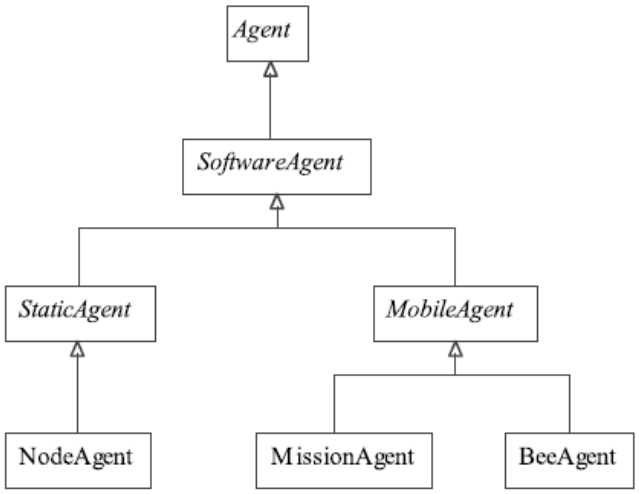


Figure 3.3: Class Diagram presenting the agents' inheritance tree.

Static Agent: This kind of agent is aimed to stay fixed in the sensor nodes as responsible for providing information about its node to the mobile agents that visit it. It provides an interface to the sensing and communication devices so that the other agents can perform their tasks, by means of exchanging messages. This method provides access to the other agents and to the local platform resources offered by the corresponding sensor nodes. It represents the sensor nodes themselves, and for this reason this agent is called ***NodeAgent***. It is important to notice that in a broader perspective, the nodes that host a nodeAgent may not be static. Even that the contribution presented in this chapter considers static sensor nodes that do not have actuators that provide mobile capabilities, a sensor node may be displaced by humans or non-intentionally, so it is important to make it clear that static is the agent in the node, but not necessarily the node itself. Even considering a mobile sensor node, this same statement about its nodeAgent holds, i.e. the nodeAgent represents the node and does not move from the node, but it follows its movements.

Mobile Agent: There are two types of mobile agents as presented below:

MissionAgents are mobile agents that represent missions and are responsible for carrying, disseminating, allocating and performing those missions in a MA defined by the mission. Their role is to take a mission to the sensor nodes in the network, according to the MA specified in the mission, disseminate it through the nodes where the mission should be performed, decide about which nodes should participate in the mission and then perform it.

BeeAgents are mobile agents responsible for distributing information about sensor nodes' status from specific regions to other locations within the MA that contain nodes that may have useful characteristics in relation to the mission requirements. Their usage provides context awareness of the conditions in the mission area to improve the mission allocation decision results. Its name is given due to the fact that their patterns of movements are inspired by the movement of bees flying from flower to flower, and by an analogy to the pollination process performed by bees distributing pollen. This analogy will be further detailed when the use of beeAgents is explained.

3.2 Mission Dissemination

The mission dissemination is performed by missionAgents, and is defined by the way these agents decide about their movement through the sensor nodes in the network. It works as follows: a missionAgent (or a group of missionAgents when a mission should be performed in different locations of the network) takes the description of the mission that is first to be injected into the network. The missionAgent then migrates from node to node towards the location specified in the mission, the Mission Area (MA), as the movement described in Section 2.2.1. Figure 3.4 shows a schematic view of the missionAgent's migration towards the MA, in which Figures 3.4a and 3.4b show the injection of the missionAgent with its mission into the network, while Figures 3.4c and 3.4d show its migration until it arrives at the MA.

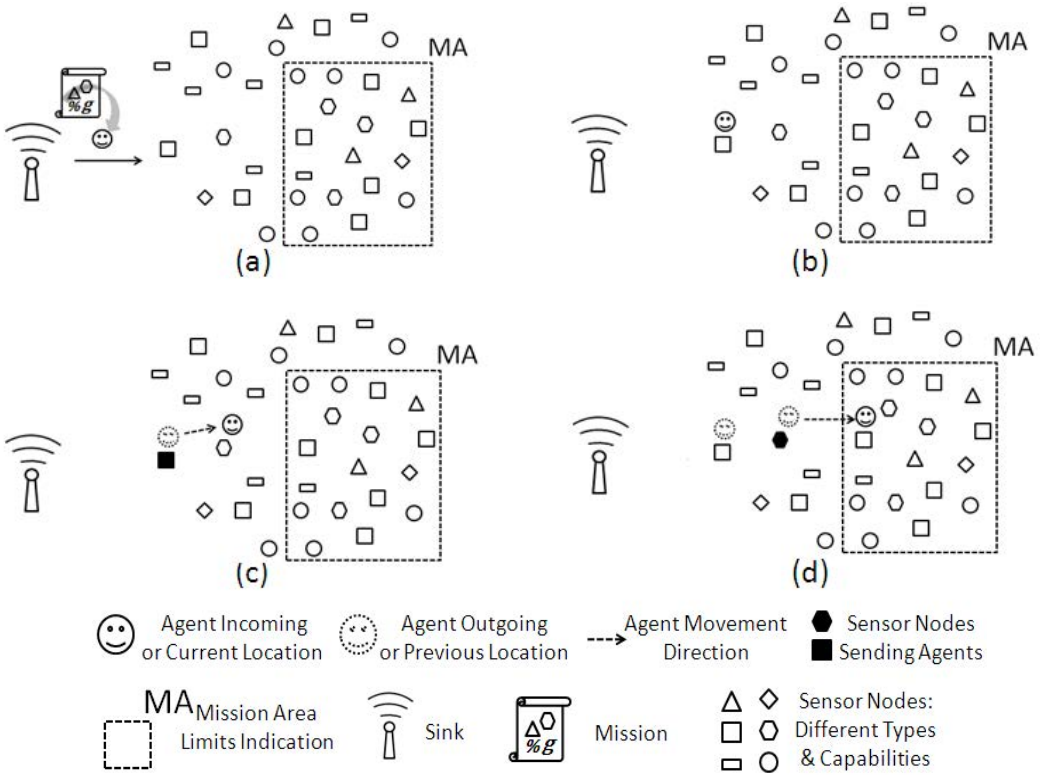


Figure 3.4: MissionAgent moving towards the TA: (a) and (b) illustrate a missionAgent being injected into the sensor network; (c) and (d) illustrate a missionAgent ***migration***.

This first agent's movement in the network is simply performed by a comparison between the current hosting node position and its previous node position. An important assumption is that the sensor nodes have the information about their own positions, which is possible by means of a GPS (PARKINSON; SPILKER, 1996) device or any other positioning mechanism, such as algorithmic solutions (DOHERTY; GHAOUI; PISTER, 2001) (NICULESCU; NATH, 2003a). All nodes within the communication range of the sender node will receive the messages sent; thus when a missionAgent is being sent, it arrives at all the neighbours of the sender node. However, only nodes closer to the MA are supposed to participate in the forwarding of the missionAgent. When arriving at a node during a migration, a missionAgent compares its current and

previous positions. If it is closer to the MA, it continues migrating, otherwise it is simply discarded. Duplications of a missionAgent may occur during a migration performed in this way, as more than one receiving node may be closer to the MA in relation to the sending node. However, as these nodes will be close to each other, a group of nodes in the direction of the MA will be able to detect that they have received the same missionAgent more than once. In this case, if the missionAgent has already been passed by another node in the direction of the MA, it is just discarded. Otherwise, if duplicated copies of the missionAgent take slightly different directions towards the MA, they will not harm the mission dissemination, but will contribute as a redundant mechanism with a negligible overhead. Figure 3.5 presents a flowchart describing this process.

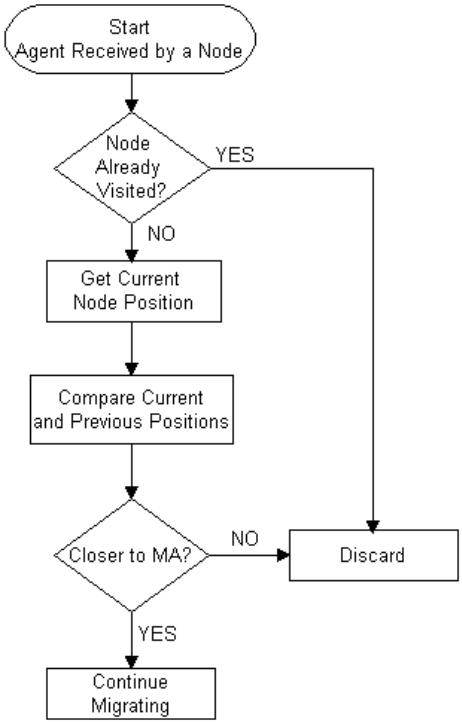


Figure 3.5: Flowchart describing the decision process for the missionAgent's ***migration***.

Arriving at the first node that can be part of the group of nodes possible to engage in a mission, i.e. a node inside the MA, the missionAgent has to reach all the nodes that possibly can be part of the mission execution. To do this, the missionAgent performs different movement actions in relation to the migration action explained above. Instead of just migrating from one node to another, it creates copies of itself (clones) which are broadcasted to all neighbour nodes of the sender node. If the sender node is eligible to perform the mission, one of these created clones is left in this node. This action that keeps a clone in the sender node and sends one of the other clones to the neighbours is called ***clone***, following the concept of agent cloning presented in Section 2.2.1. If the sender node is not eligible to perform the mission, no copy of the agent (no clone) is left in the sender node. Thus this action is called ***migrate-clone***. In Figure 3.6 these two

movements are presented, starting from the entrance of the missionAgent in the MA, depicted in Figure 3.6a. A migrate-clone is presented in Figure 3.6b and a clone is presented in Figure 3.6c. The actions clone and migrate-clone are repeated until all pertinent nodes in the MA, i.e. nodes eligible for the mission, have got a "geocasted" copy of this agent.

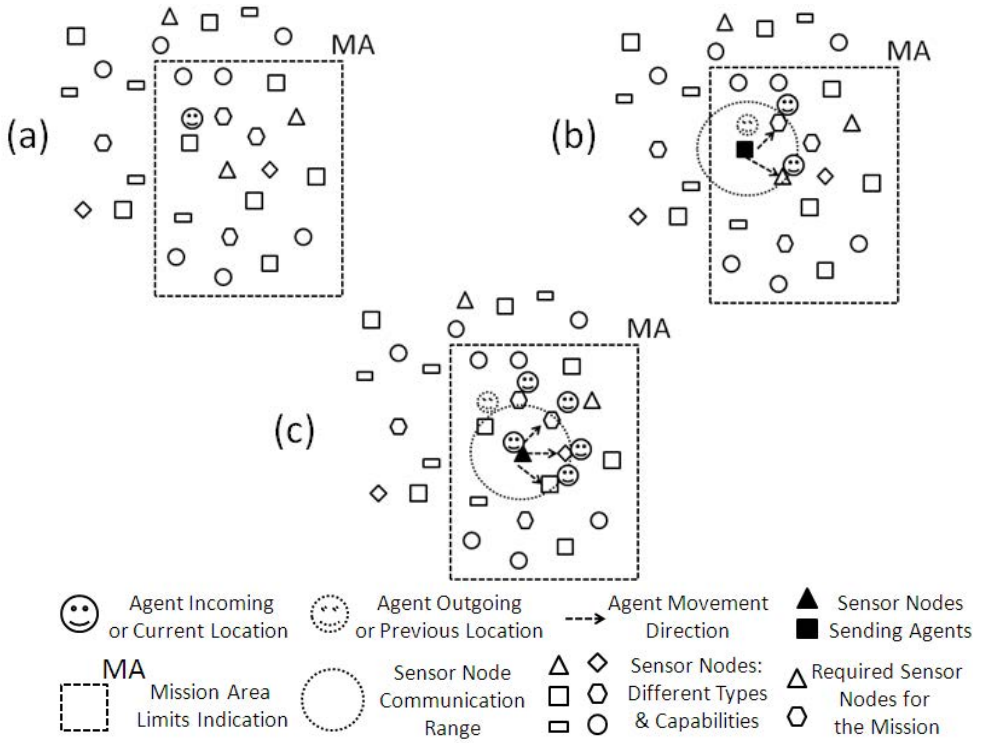


Figure 3.6: Agent movements inside the MA: (a) initial node inside the MA; (b) ***migrate-clone***; (c) ***clone***.

The decision about which kind of movement action a missionAgent will take inside the MA to proceed with the mission dissemination is done by the analysis of the node that is currently hosting the agent, by means of an exchange of information with the nodeAgent. Figure 3.7 presents a UML sequence diagram depicting this message exchange process. It is noteworthy to mention that the messages exchanged by the agents used in this work follow the semantics specified in the FIPA Agent Communication Language (ACL) (FIPA, 2002a).

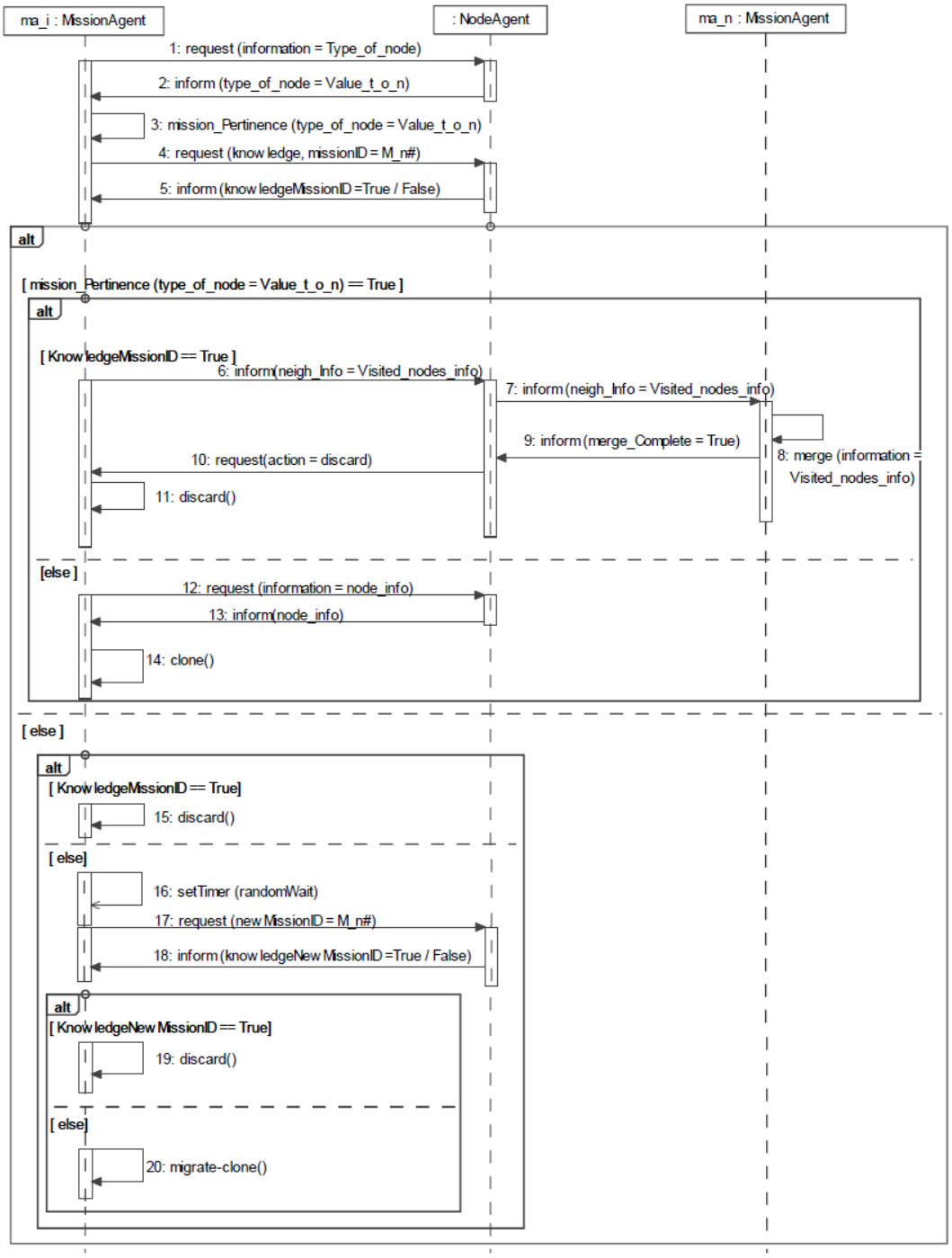


Figure 3.7: Interaction between the agents while a mission is being disseminated.

In the first step of this message exchange process, the missionAgent acquires information about the type of the node (1 and 2 in Figure 3.7), so that it can decide if the node is eligible or not to perform the mission that it is carrying (3 in Figure 3.7). Then it requests if the nodeAgent has knowledge about its mission, identified by its missionID, and the nodeAgent informs if it knows about that mission or not (4 and 5 in Figure 3.7). If the sensor node is eligible to perform the mission, and if the nodeAgent has

knowledge about that mission, it means that a similar missionAgent is already in that node, i.e. another missionAgent, a clone representing the same mission, has arrived to this node. In this case the most recently arrived missionAgent provides information about the neighbour nodes that it previously visited to the nodeAgent of the current node (6 in Figure 3.7). Then the nodeAgent takes this recently acquired knowledge and informs this to the missionAgent that is already in the same node (7 in Figure 3.7). By its turn, this other missionAgent merges the received information to its own and informs the nodeAgent when it completed the merging (8 and 9 in Figure 3.7). The recently arrived missionAgent is then requested to be discarded and performs this action (10 and 11 in Figure 3.7). Otherwise, if the nodeAgent does not have knowledge about the mission, it means that no similar missionAgent visited this node. In this case, the missionAgent requests the node's status, e.g. energy level, sensor devices capabilities and accuracy, among others (according to mission criterion – the goodness), and then decides to clone (12 to 14 in Figure 3.7). To limit the size of the data that an agent is carrying about other nodes eligible to perform the mission, the goodness status information of the visited eligible nodes is processed and an average is calculated before the agent clones. In order to do this, the agent keeps information about the number of nodes visited so far that contributed with this accumulated average, such that the average can be updated with the contribution of the recently visited node. Besides this average, it also carries the raw goodness value of the last visited node.

If the node is not eligible to perform the mission and the nodeAgent already has knowledge about this mission, the missionAgent just decides to do a discard action (15 in Figure 3.7). In turn, if the nodeAgent does not have such knowledge, i.e. it is the first time the node receives an agent carrying this mission, the missionAgent waits a random time interval by setting a timer (16 in Figure 3.7) and requests information about a possible arrival of a similar missionAgent, i.e. a missionAgent with the same mission, by exchanging messages with the nodeAgent (17 and 18 in Figure 3.7). If a similar missionAgent arrived during this time, it means that another node in the neighbourhood already sent a clone of the same missionAgent (this switches the value of the `knowledgeNewMissionID` for that mission to a value equal to true in the current node), thus the one in the current node will be discarded (19 in Figure 3.7), otherwise it decides to migrate-clone (20 in Figure 3.7). This random time waiting and then checking about the arrival of a similar missionAgent (16 to 18 in Figure 3.7) avoids unnecessary redundant forwarding of agents carrying the same mission, thus saving energy resources.

Once all eligible sensor nodes available and reachable in the MA have received a copy of the missionAgent, the mission is considered disseminated.

In order to keep the missionAgent inside the bounds of the MA, a simple location consistency check is performed by the missionAgents while they perform clone or migrate-clone actions. When a copy of an agent being disseminated in the MA, arrives at a node after these two types of movement actions, it checks if this node has its coordinates within the MA limits. If this is the case, then it proceeds as explained above. Otherwise, it just decides to discard. This process is similar to the one presented in the flowchart of Figure 3.5.

3.3 Mission Allocation

The proposed solution for mission allocation has the goal to provide a fully decentralized decision mechanism, to avoid extra overhead in terms of communication due to unnecessary exchange of control messages. Thus, the idea is to take as much advantage as possible of the information collected by missionAgents during the dissemination phase and use it to decide about the mission allocation.

3.3.1 The Decision Procedure

Once a missionAgent arrives at a node inside the MA and decides to clone (call number 14 in Figure 3.7), the clone of the agent that stays in the node and that not is forwarded starts a timer. By the expiration of this timer, the agent initiates the allocation decision procedure. The timer is set to the estimated time to be elapsed until all nodes in the MA are visited by the missionAgent to disseminate a mission. After the timer expiration, the mission is considered disseminated. The choice of this duration is done by considering both the number of hops that a missionAgent should take to cross the MA from its entry point at arrival to the most distant node and the communication range of the sensor nodes. To avoid premature start of the decision procedure in the first nodes visited by the missionAgent, the timer is set to a value equal to twice the calculated value (in order of seconds).

By the expiration of the timer, the missionAgents in each node consider that the mission is disseminated. This means that all nodes in the MA should have been visited by a clone of the missionAgent at least once, and, as there are no more movements, there is no additional information about neighbours coming to any node via the clones. Thus, the decision procedure, using information about the nodes in the neighbourhood, can start.

In each node, the missionAgent will decide locally if it engages the hosting node in the mission or not, using the goodness information of the node, plus the information about the node's neighbours, which was acquired when these neighbours transmitted the missionAgent during a clone or migrate-clone agent movement action. This is the reason why the timer mentioned above is used to trigger the start of the decision procedure only after at least one missionAgent has surely arrived to any node in the MA.

This local decision about the engagement of the node includes a weighted probability calculation. By receiving information about its neighbours, the missionAgent makes a generalization of the resources and capabilities available in its neighbourhood, thus constructing an approximation of what exists in the entire MA. From such reasoning, it assumes that the amount of requested nodes in its neighbourhood has to be proportional to the one required in the mission directives for the entire MA (fifth element in the mission structure, see Figure 3.2).

Each missionAgent adds the goodness values received from its neighbours to the one of its hosting node and calculates the contribution of its node to this result, corresponding to the probability of the node's engagement in the mission, according to the following:

$$prob_i = \begin{cases} \dfrac{g_i}{\sum_{j=i}^{n} g_j} \cdot p \cdot n, & \text{if } prob_i < 1 \\ 1, & \text{otherwise} \end{cases} \quad (3.1)$$

where: g_i is the result for the goodness calculated for node i; p is the percentage of the nodes capable to accomplish the mission that should engage on that mission; and n is the number of neighbours with the required sensing capabilities and that are taken into account in the decision procedure executed at node i.

Each node will then randomly decide whether to engage or not in the mission, based on the value obtained in (3.1). The result $prob_i$ is thus compared to a random value (***rand***) between 0 and 1 obtained from a random function with linear distribution. If $prob_i$ is bigger then ***rand***, the node engages in the mission, otherwise it does not engage. Notice that the random value used in the comparison could also be a fixed value, which if closer to 1 would decrease the probability to the nodes engage in the mission while values closer to 0 would increase this probability. The use of the random value aims to avoid bias the decision.

Figure 3.8 presents a flowchart describing this procedure to decide about the node engagement.

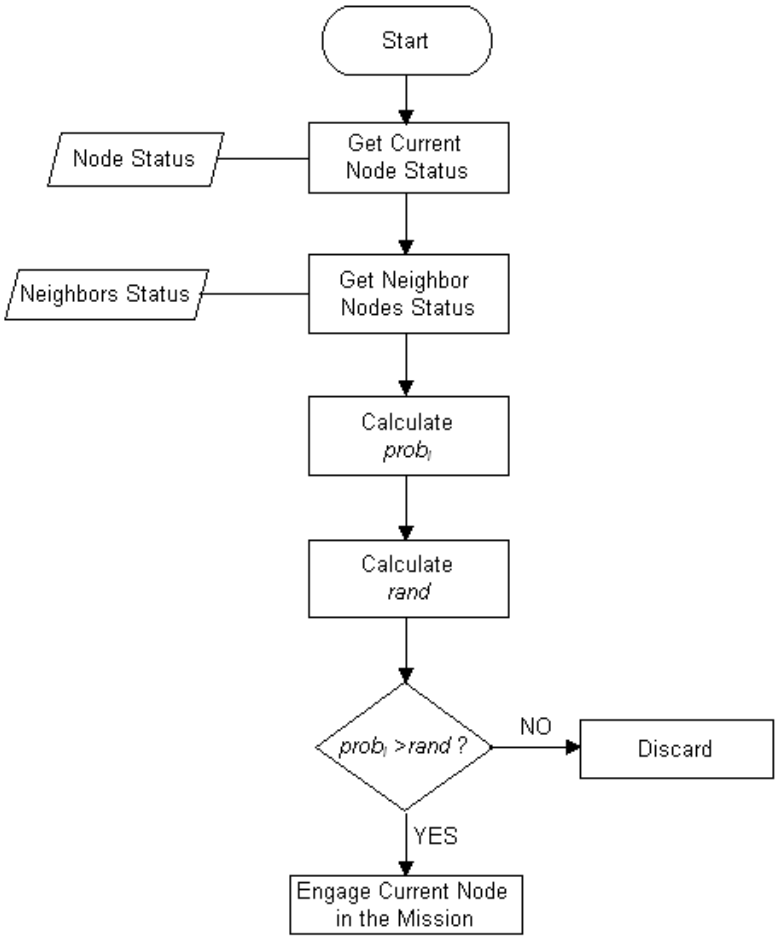


Figure 3.8: Mission Allocation Decision Procedure.

3.3.2 Overcoming Unfavourable Situations Caused by Unbalanced Nodes' Distribution

An assumption made, influencing the efficiency of the above described decision procedure, is that the nodes are randomly and uniformly distributed in the area where the system is deployed, and thus also in the MA. This also models a MA equipped with a heterogeneous set of sensor nodes, with respect to their abilities to perform a given sensing mission depending on the requirements of the mission, i.e. their goodness. For instance, a mission that requires nodes with a given level of accuracy or a certain remaining energy level may find sensor nodes, which are more or less suitable to perform it, well distributed in the MA.

In the example shown in Figure 3.9a, the sensor nodes are randomly distributed in the area according to their capabilities to accomplish a given mission, which are represented by the gray scale, in which lighter nodes represent those with better capabilities, thus better goodness. However, in a situation in which the nodes are not heterogeneously distributed according to their capabilities, as presented in Figure 3.9b, there is a risk to bias the decision resulting from the mechanism described in the previous section. This risk comes from the fact that, by exploring the locality considering the nodes' neighbourhood, the decision may result in the engagement of nodes with poor goodness results that are in regions (parts of the MA) in which there are concentrations of nodes with poor conditions. At the same time, nodes with higher goodness values will not be engaged in other regions in which they are surrounded by other nodes also with high goodness levels. This will result in leaving a number of nodes that could be used in the mission unused, while engaging nodes that are not the most appropriated ones. An example of distribution in which this situation may occur is shown in Figure 3.9b. Such unpaired distribution can occur if a specific part(s) of the network has (have) too much activity, for instance if it (they) is (are) overused for routing in the case in which the energy level is the considered condition. This may happen if the same route is overused to deliver messages to a sink node, or if the network is performing several missions, and some of them overuse the sensor nodes in specific parts of the MA (regions in the MA), while leaving other parts with idle unused nodes.

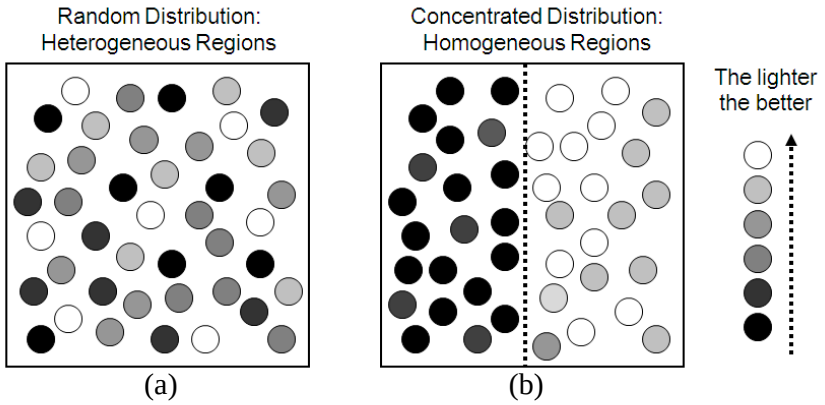


Figure 3.9: Sensor nodes distributions: (a) Random Distribution (Heterogeneous or Mixed); (b) Unbalanced Distribution with Homogeneous Concentrations.

To avoid inefficient mission distribution in the cases of existence of regions with concentrations of nodes with similar status, the missionAgents have to be aware of these regions, so that this can be taken into account in the decision making procedure. However, to address this issue, two problems have to be solved: the first one is to detect the existence of regions with concentrations of similar nodes, and the second one is to alert the missionAgents in other nodes in the MA about such concentrated regions.

The proposal is to solve the first problem by means of a simple calculation of the average and the standard deviation of the goodness values of the neighbour nodes and then comparing this average with the accumulated averages brought to the node by the missionAgents during the mission dissemination phase. An indication of a concentration of similar nodes is given by the standard deviation of the nodes' goodness in the neighbourhood. If it is too small, below a given threshold, it means that the nodes individually have goodness values very close to the average of the considered group of neighbours, and thus close to one another which characterizes the region as homogeneous. Otherwise, if the standard deviation is above the threshold, the region is not considered as having a concentration of similar nodes but instead being characterized as heterogeneous. However, even with an indication that a particular region within the MA is homogeneous, it may happen that all nodes in the MA have a similar characteristic, i.e. the whole MA is homogeneous, which is very probable when, for example, the network has been recently deployed (all the nodes in the network have high goodness values) or when the network is close to the end of its lifespan (all nodes have low goodness values). This means that there are special circumstances that need to be considered.

Therefore, even if the first test indicates that the region is homogenous, a second check is done by comparing the accumulated goodness average that came with the missionAgent during the mission dissemination phase. If this average is close to the neighbours' goodness average, i.e. the difference between them is smaller than a threshold, then it is an indication that all nodes in the MA are homogeneous, i.e. all the nodes have high or low goodness values. Otherwise, if the averages' are not close, then the MA is not homogeneous and a region of concentration has been identified.

For this type of analysis to make sense it is assumed that, during the mission dissemination, the accumulated goodness value carried by the clones of a missionAgent has information about nodes in different parts of the MA. However, it is very probable that, for nodes close to the entrance point where the missionAgent arrived at the MA, this comparison will not be so meaningful, as the missionAgent did not have yet the chance to visit many nodes before. But, besides this borderline situation, for the other nodes (the general case) it is reasonable that such a comparison is worth to be done, thus giving the opportunity to this type of analysis to work properly. Figure 3.10 presents the flowchart describing this algorithm, aimed to discover regions with concentration of nodes with similar properties (homogeneous regions).

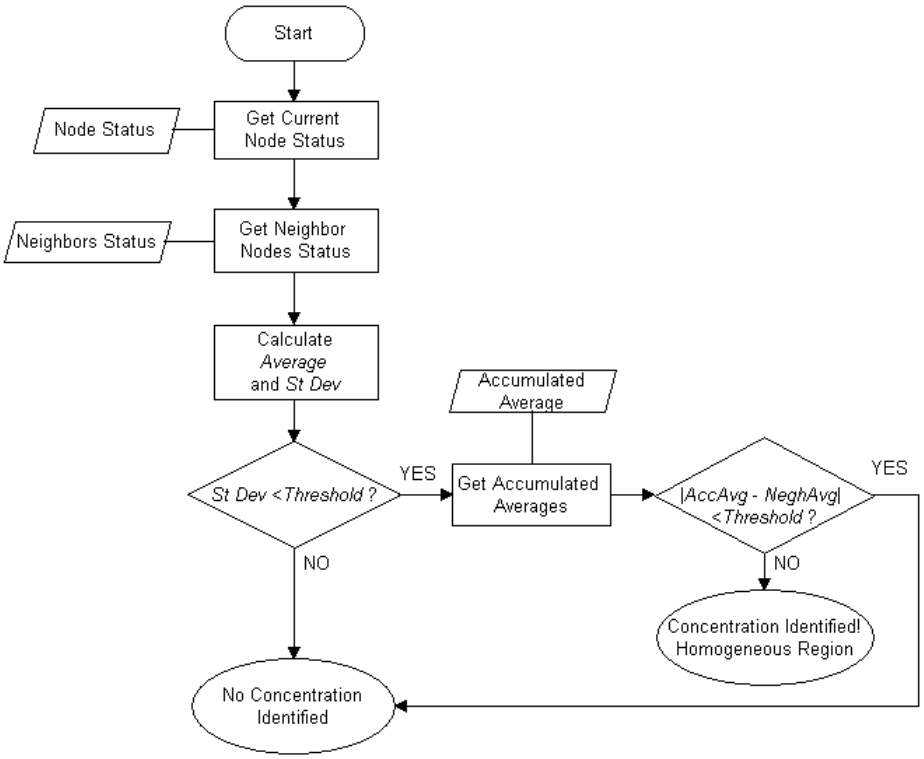


Figure 3.10: Identification of regions with concentration of similar nodes (homogeneous regions).

Having solved the first problem, namely the identification of regions with similar nodes, the next step is to inform other nodes in the MA about the existence of such homogeneous regions. Being aware of the existence of such regions and their characteristics, other regions with complementary characteristics can contribute with their nodes' goodness values in the calculation while the missionAgents take the decision to engage or not a node in a mission. To provide this feature, the second mobile agent introduced in Section 3.1.2 is used, i.e. the beeAgent. Its name is given by the way bees fly from a flower to another in their search for nectar and pollen. When collecting nectar or pollen from the flowers, bees, by using their sense of direction, are capable to rapidly recognize that a given flower was recently visited. Having such information, the bees are able to select non-visited flowers to continue their movement and collect food. Moreover, while visiting flowers, bees contribute with the plants' reproduction mechanism by a pollination process called entomophily, in which they take pollen from a flower to another (PARTAP, 2011). In a similar way, the beeAgents are capable to recognize nodes already visited while moving through the network, so that they follow a path of non-visited nodes, and also transport information about the group of nodes with similar characteristics from where they came to other nodes, as bees carry pollen from flower to flower.

Considering a sensor network with uniform distribution of nodes, it is expected that, in average, the nodes will be distributed uniformly according to their characteristics, as the example presented in Figure 3.9a. However, due to a number of circumstances, such as hot spots created by overused routes or higher incidence of events in a given part of the area where the sensor network is deployed, it is reasonable to assume the existence

of regions with concentration of nodes with similar characteristics, such as the example presented in Figure 3.9b. However, it is probable that other similar situations not too extremely adverse as presented in Figure 3.9b occur. Such situations represent regions of concentrations which are surrounded by heterogeneous sensors, as illustrated in Figure 3.11. This requires a more efficient way than a conventional flooding to inform the sensor nodes about these complementary regions. The beeAgent is then responsible for spreading information about such concentrated regions, in an attempt to improve the mission allocation. The reason is that the probabilistic decision explained before assumes diversity among the nodes' goodness values, and situations with node concentrations biases the results.

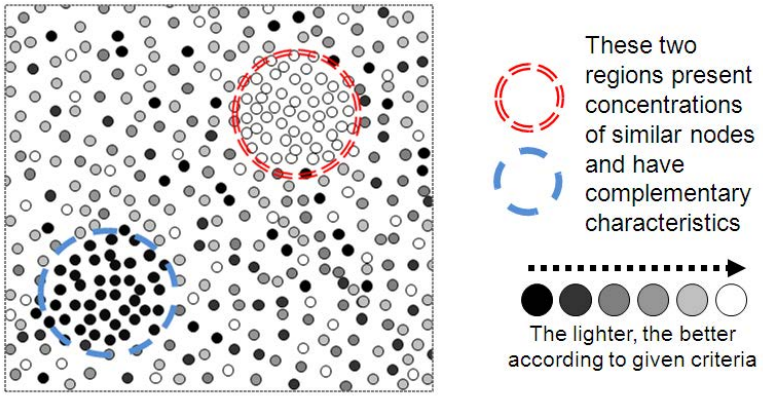


Figure 3.11: Example of regions of concentration of similar nodes immersed in the network.

The creation and sending of beeAgents is decided by the missionAgents for situations in which they find regions of concentration in the network, according to the identification algorithm described above. This information about whether the region in which the sensor node is located has a concentration of similar nodes or not is shared with the nodeAgent, which is responsible for informing incoming beeAgents that arrive from complementary regions about the local situation. BeeAgents will be sent from one or more nodes in the concentrated regions, depending on the relation between the communication range and the size as well as shape of the region. Then they follow a semi-random walk through the network, by selecting random directions so that they are forwarded in a way that do not lead them back to the region where they came from, in a way similar to how bees recognize flowers already visited. This is a migration action similar to the one performed by the missionAgents explained in Section 3.2. However, instead of having a precisely defined destination as the MA in the missionAgent's migration, the direction followed by the beeAgent's is randomly selected inside the MA.

When beeAgents find regions with complementary characteristics, they disseminate information about the existence and properties of their respective source regions by means of a clone movement action through the eligible nodes in these regions. This delivery of information to complementary regions is comparable to the bees' pollination process mentioned above. If the beeAgent does not find any complementary region after a given number of hops or after that it reaches a limiting edge of the MA, it is just discarded by the last visited node.

After identifying the homogeneous region, before a missionAgent decides to send a beeAgent, it first waits a random time interval and then checks if a missionAgent in another node in the vicinity had already sent a beeAgent telling about this same region of concentration. If, after this waiting time, a beeAgent was already sent, there is no need to send another beeAgent carrying the same information, otherwise, the beeAgent is sent. One neighbour node is chosen to forward the beeAgent, according to the selected random direction. Before the beeAgent forwarding proceeds from this node, it is performed a check if a beeAgent informing about the same region was already forwarded by another neighbour node. Figure 3.12 presents the algorithm regarding the decision to create and send a beeAgent.

Once a beeAgent is sent, the missionAgent is aware that it is possible that another beeAgent from a complementary region may arrive. So, it should wait a certain amount of time before starting the decision process. On the other hand, if the node is not in a homogeneous region, the missionAgent can directly start the decision procedure after running the algorithm to identify homogeneous regions. The complete flow-chart with the whole process explained above is presented in Figure 3.13.

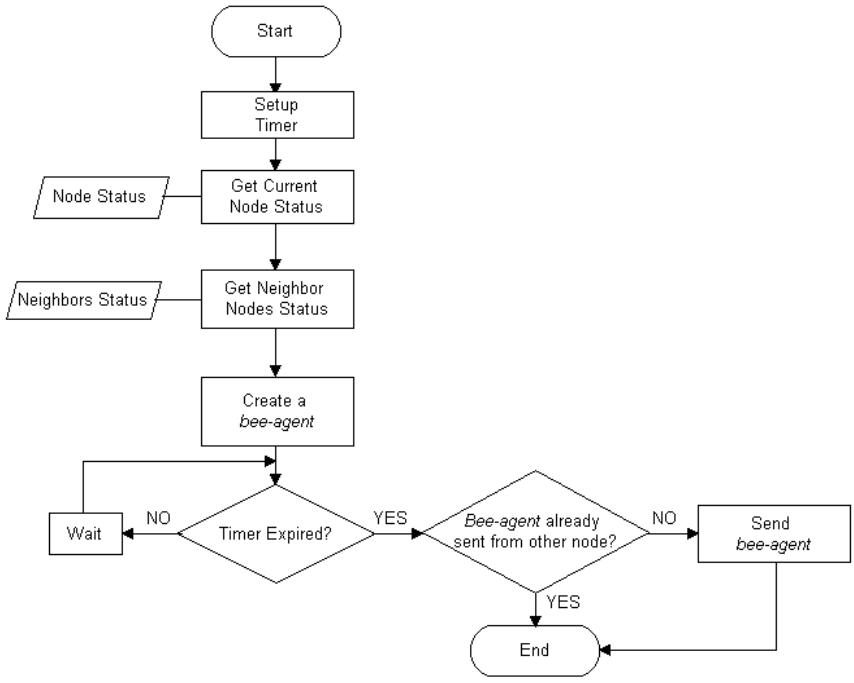


Figure 3.12: BeeAgent sending algorithm.

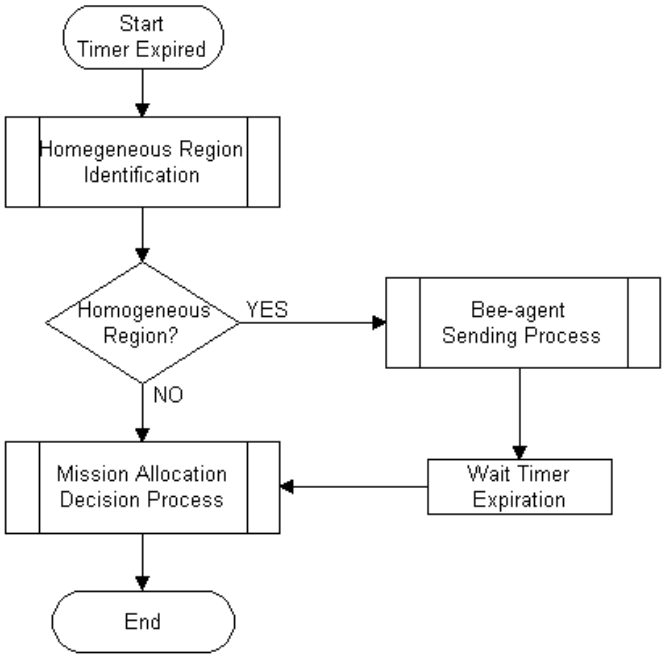


Figure 3.13: Complete process to perform the mission allocation.

3.4 Experiments and Results

This section presents a set of experiments performed to evaluate the performance of the mission dissemination and mission allocation mechanisms proposed for static WSNs proposed in this work. The experiments were conducted in the form of simulations using GrubiX simulation tool, in which the proposed agents are implemented as Java objects.

Different simulation setups were used to highlight the usefulness of the proposed solution and to measure the benefits and overhead of the beeAgents.

3.4.1 Simulation Setup

Basic Setup

In the performed simulations, the mission is small enough and fits in a tiny missionAgent that can be sent using just one communication packet.

The mission area, MA, has dimensions 5 Km $\times$ 5 Km, in which 8000 sensor nodes are randomly positioned according to an independent uniform probability (homogeneous Poisson point process (ROSS, 2007) in two dimensions, which generates a geometrical random graph). This density and distribution results in a probability of 71% that the nodes in the network form a connected graph (BETTSTETTER, 2002), assuming a communication range of 100 meters, and a probability of 100% with a communication range of 200 meters, which were the values used in the performed simulations. The condition about the network connectivity is important due to the fact that the nodes have to be connected to the rest of the network in the MA to receive the mission being disseminated; there should be at least one link that individually connects

each node to the rest of the network, i.e. there are no isolated nodes. Considering this restriction, for the simulations with 100 meters communication range, only those instances of the nodes' distributions that formed a connected graph were used.

The sensor nodes each have one of four possible types of sensors: humidity, temperature, vibration and luminosity. These capabilities are equally distributed among the nodes.

Simulated Mission

As the purpose of the contribution presented in this chapter is to present a solution for the sensing mission dissemination and allocation, just the first five fields of the mission structure (see Figure 3.2) were effectively used to specify the mission.

The mission that is requested to be assigned in the simulation requires luminosity sensing capability and has its evaluation criterion based on the sensor device accuracy and on the remaining energy level. The mission specifies that the nodes that have better sensor, for example with better accuracy, and higher energy levels are to be engaged in the mission and that both parameters have the same importance, such that the goodness function can be written as:

$$\boldsymbol{g} = \quad \cdot \boldsymbol{ac} + (1 - \quad) \cdot \boldsymbol{e} \tag{3.2}$$

where $\boldsymbol{ac}$ is the sensor accuracy and $\boldsymbol{e}$ is the remaining energy level. The value α is a tuning factor that determines the importance of each parameter in the goodness calculation. In this case, it has value equal to 0.5, as both parameters (accuracy and energy) have the same importance.

The mission requires that 50% of the eligible nodes, i.e. those that have the required sensing capabilities, are engaged in its accomplishment. From the total number of nodes (8000), 2000 ones have the required sensing capability to perform a mission (they are thus eligible), i.e. luminosity sensing capability, and 50% of them (1000 nodes) should be engaged in the mission according to the criterion established as a mission requirement.

Setup Variations

From the common setup parameters presented above, two different variations were derived. In the first one, each node starts the simulation with different sensor accuracy and different energy levels available. The sensor accuracy and energy levels are randomly distributed according a uniform distribution in the same range, from 10% to 100% of their full capability. Thus, the node distribution in the area in relation to their capabilities to perform the mission, i.e. their goodness, is similar to the one presented in Figure 3.9a. This first variation will be called "setup-Random" in the presentation of the results.

The second variation represents a scenario in which the network is divided in two areas, according to the distribution of the eligible nodes for the mission. Half of the nodes eligible to the mission are concentrated in one part of the area, with energy and sensor device accuracy level varying between 90% and 100%, while the rest of the eligible sensors are concentrated in the other half of the area, having at most 10% of remaining battery and 10% of the sensor device accuracy. This provides a distribution with concentration of nodes similar to Figure 3.9b. In the presentation of the results, this second setup will be called "setup-Conc" (for "setup-concentration").

For the two setups described above, two sets of results were obtained, one using the communication range of 100 meters for the sensor nodes and the second one using a range of 200 meters.

Another variation of the performed simulations was in relation to the usage of the beeAgents. Two types of simulations were performed, one with the beeAgents (running the complete mission allocation procedure presented in Figure 3.13) and without beeAgents (running the mission allocation procedure as presented in Figure 3.8 only). For each setup variation, a set of 100 runs were performed. Table 3.1 summarizes the setup variations.

Table 3.1: Simulation Setup Variations

Setup Variation	Node Distribution	Communication Range	Program Variation (with or without beeAgent)
Setup-Random-100-NB	Random	100	No BeeAgent (NB)
Setup-Random-200-NB	Random	200	No BeeAgent
Setup-Conc-100-NB	Concentrated	100	No BeeAgent
Setup-Conc-200-NB	Concentrated	200	No BeeAgent
Setup-Random-100-WB	Random	100	With BeeAgent (WB)
Setup-Random-200-WB	Random	200	With BeeAgent
Setup-Conc-100-WB	Concentrated	100	With BeeAgent
Setup-Conc-200-WB	Concentrated	200	With BeeAgent

3.4.2 Results and Discussion

The results present three metrics aimed to assess the efficiency of the proposed approach. The first metric assesses the efficiency of the mechanisms to reduce the energy consumption. Considering that communication is the major energy consumer in WSNs, this metric evaluates the reduction in the data communication, as compared with an ordinary flooding-based mechanism presenting the maximum number of packets needed to disseminate the mission among the nodes. This result also provides information about the communication overhead, compared to the cost for the optimal case, of the proposed approach. The second metric gives the average number of nodes engaged in the disseminated mission, which is compared to the desired number of nodes to be engaged. The third metric gives the average goodness of the engaged nodes, as the goal is to engage those nodes that have better goodness.

An important rationale for the proposed approach is to assign missions to the sensor nodes with low overhead due to transmission of control messages. To achieve this goal, the proposed agent-based approach disseminate and allocates the mission in a decentralized way, in which the only overhead is caused by the mission dissemination and by the spreading of information about regions with similar nodes concentrations, if

such regions are identified. Figure 3.14 presents this communication cost, in which the average number of sent packets carrying agents is shown for each set of simulation runs, corresponding to each setup variation described above. To give support to comparisons, the optimal solution based on a search of the global state of the simulations and results of an ordinary flooding-based solution are also presented. The optimum solution is implemented by storing the information about the type of the sensor and the respective goodness value for each individual sensor node, as well as their position. Based on the criteria defined in the mission, the more suitable sensor nodes are selected from this global view of the network, i.e. only the 2000 eligible nodes in this simulated scenario. The flooding-based considers that all 8000 sensor nodes forward the mission when they receive it.

Notice that Figures 3.14b and 3.14d present results for two variations of the optimal and flooding solutions. The reason for this is due to the fact that the first optimal solution does not consider the identification of the regions of node concentration and is thus comparable to Setup-Conc-100-NB and Setup-Conc-200-NB, which holds for the first flooding results too. On the other hand, the second results for the optimal solution (Optimum-B) considers the dissemination of the information about regions of concentration, and is thus comparable to Setup-Conc-100-WB and Setup-Conc-200-WB, and the same holds for the second flooding results (Flooding-B). Optimum-B solution calculates the shortest path between regions of concentration, i.e. the minimum number of nodes that links the regions, and considers that the beeAgent follows this path. Then the beeAgent is disseminated in the region following clone movements through eligible sensor nodes, as explained in Section 3.3.2. Notice that the results for the Optimum, Flooding and Flooding-B do not present error bars because they have always the same value. This result is due to, in the case of the Optimum, only the 2000 eligible nodes forward the missionAgent, while in the Flooding, every node forwards the missionAgent during the dissemination, and in the Flooding-B, every node forwards the missionAgent, and then all of them forward the beeAgent. As the dissemination of the beeAgent in the Optimum-B varies according to the nodes distribution for each run, the average result is presented with the respective standard deviation.

It is possible to observe that the number of messages sent for the setups with communication range of 200 meters is lower than those for the corresponding setups with 100 meters range. Due to the nature of the broadcast, i.e. sending a single packet enables the reception of its information in all nodes located within the communication range of the sender node, this difference can be explained by the lower probability that the nodes will forward a packet containing an agent in the 200 meters range setups due to the longer range in this case, which enables a packet sent by a sensor node to reach a greater number of other sensor nodes. This means that a sensor node has more neighbours in the long range case and this impacts the number of packets that need to be sent according to the mission dissemination procedure explained in Section 3.2. As described there, if an agent is prepared to migrate-clone from a node and this node receives a similar agent, the agent does not proceed with its movement. This decision avoids unnecessary redundant transmissions, as confirmed by the results shown in Figures 3.14c and 3.14d, i.e. it reduces the total number of sent packets.

The results presented in Figures 3.14a and 3.14c also reveal that the variations with random distributions (Setup-Random-100NB, Setup-Random-100WB, Setup-Random-

200NB and Setup-Random-200WB) are not affected by the beeAgent, because in fact it is not used (even in the WB setup variations, because no region of concentration is identified), so there is no increase in the number of packets sent due to its transmission. Similar results can also be seen for the variations with node concentration that do not use the beeAgent (Setup-Conc-100NB and Setup-Conc-200NB). This shows the consistency of the process in correctly recognizing regions of concentration. The increased number of packets sent is then observed in the variations with node concentration and in which the beeAgent is used (Setup-Conc-100WB and Setup-Conc-200WB). These results are consistent with the expected ones, as the usage of the beeAgent implies more packets being communicated in the network, besides those used for the mission dissemination. However, considering the high concentration of the nodes in the simulated concentrated scenario, i.e. two very distinct regions of concentration, the increase of around 30% in the overhead due to the use of the beeAgent is fairly acceptable compared to the 100% increase of Flooding-B. This statement is based on the fact that the dissemination of the beeAgent in these highly concentrated regions requires that the beeAgent be forwarded by a number of nodes to cover all the nodes in this region. Notice that this is an extreme case in which these regions are very big in relation to the entire MA. Cases in which these regions are smaller would require less communication among the nodes.

Compared to the flooding-based solutions, these results represent the contribution of the proposed approach, as no additional package besides those used in the mission dissemination are used in the whole process when no regions of concentrations are identified. In the cases in which such regions are detected, only around 30% additional packets are required to disseminate information about these regions by using the beeAgent.

Besides the importance in reducing the overhead due to communication among nodes, another important concern in WSN based systems is to search for solutions in which an optimal or close to the optimal number of resources are used. In the case considered in this paper, each mission specifies the number of sensor nodes it requires. Thus, the goal is to engage a number of nodes as close as possible to this target number. This implies that the network should avoid to use more sensor nodes than required, but also avoid to engage too few so that the mission could not be performed. The number of engaged nodes in the mission is the metric used to assess how good the approach performs in relation to this goal. Figure 3.15 presents the average number of engaged nodes for each set of runs, corresponding to each setup, compared to the target number of nodes to be engaged in the mission, i.e. 1000 nodes in this experiment.

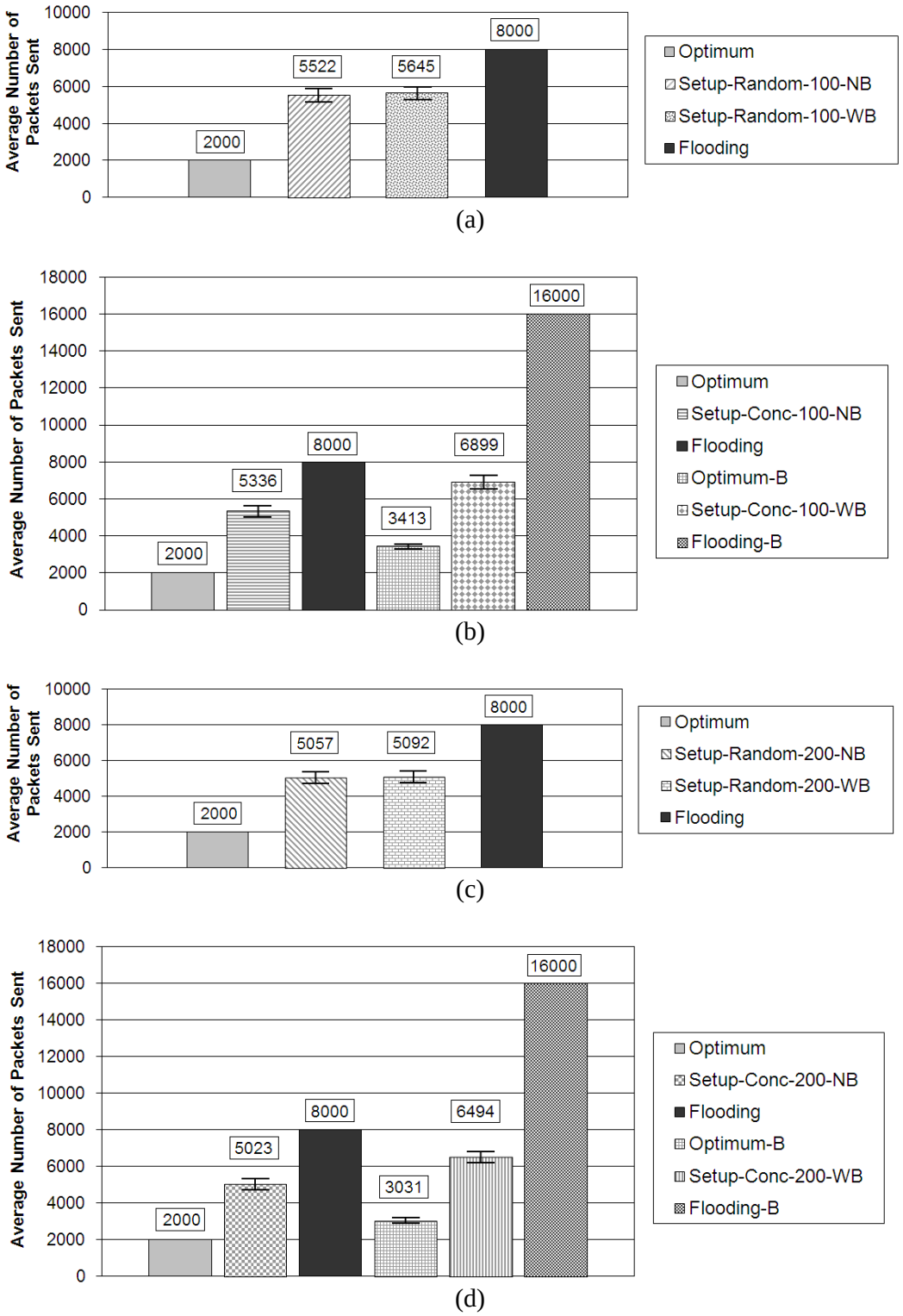
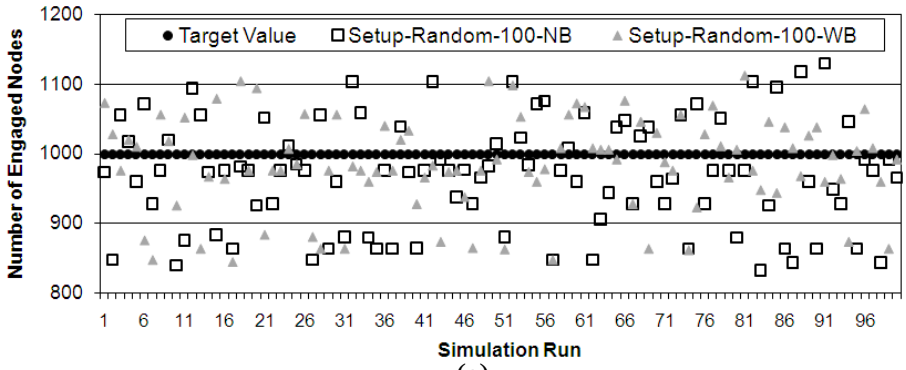
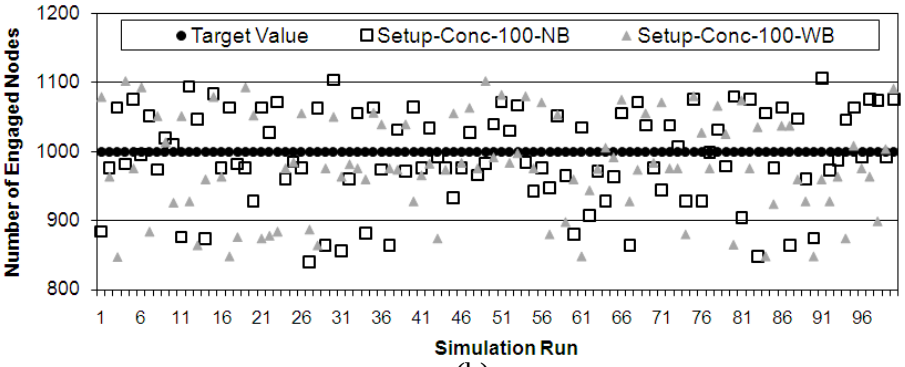


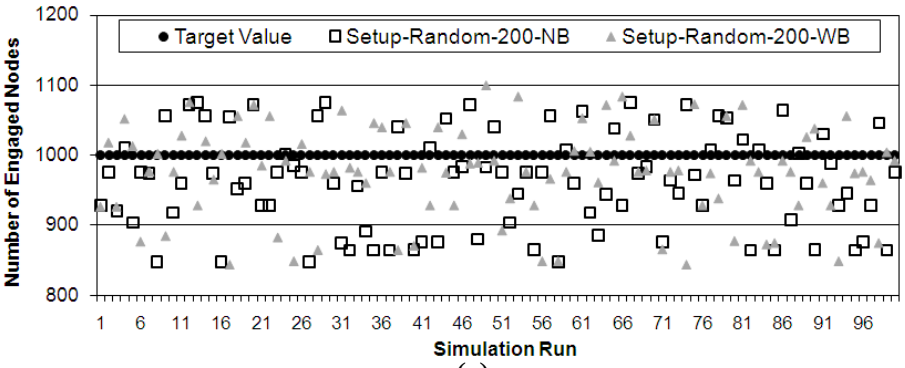
Figure 3.14: Average number of agent packets that are sent in each solution (the error bars indicate the standard deviation).



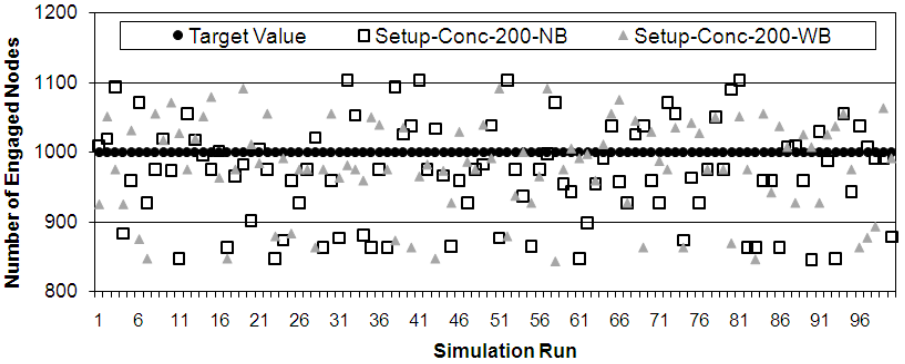
(a)



(b)



(c)



(d)

Figure 3.15: Number of nodes engaged in the mission.

The results presented in Figure 3.15 are complemented by the statistics presented in Table 3.2, from which one can observe that all setups have averages very close to each other and to the target value (1000). Moreover, the values for the standard deviations are not too high, in which the highest one was 77.55, which was observed for the measurements of the setup Setup-Random-100-NB. These results certify the robustness of the proposed approach in engaging a number of nodes that does not represent a waste of system resources, i.e. it does not engage an unnecessary big number of nodes, and at the same time it is able to perform the mission, as it does not engage too few sensors either.

A careful observer may ask why the average values presented in Table 3.2 are all below the target number. This occurs due to the result of the comparison between $\boldsymbol{prob_i}$ calculated by (3.1) and ***rand*** (see Section 3.3.1). The comparison is based on a "greater than" operator between the two values only, as presented in the flowchart of Figure 3.8. Replacing the comparison with a "greater than or equal" operator will give a higher number of engaged nodes. Additional simulations with the same setups reveal that increasing or reducing one of these values by a tuning factor can also be used to shift the results to lower or greater averages. However, since the achieved results can be considered "good enough", i.e. very close to the target value, variations of this possible tuning factor were not further explored.

Table 3.2: Number Engaged Nodes Statistical Results.

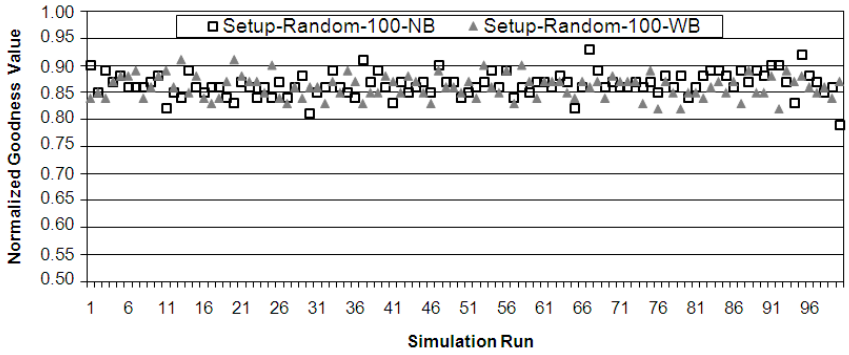
Setup Variation	Average	Maximum	Minimum	Standard Deviation
Setup-Random-100-NB	970.11	1130	833	77.55
Setup-Random-100-WB	983.70	1112	845	67.14
Setup-Conc-100-NB	993.20	1106	840	68.49
Setup-Conc-100-WB	978.36	1103	847	70.24
Setup-Random-200-NB	963.63	1075	848	68.43
Setup-Random-200-WB	975.34	1100	843	66.39
Setup-Conc-200-NB	970.46	1104	846	71.23
Setup-Conc-200-WB	978.54	1092	844	66.94

The last assessed metric provides information about how suitable the selected sensor nodes are to perform the mission. Together with the number of nodes engaged to perform the mission, this metric measures how well the user needs, as described in the mission specification, are met. It is important to engage a number of nodes close to the one specified by the mission, but of equal importance is the quality of the results that these sensor nodes can provide, which is specified by the goodness function that evaluates how suitable a sensor node is for a given mission. Figure 3.16 presents the normalized averages for the goodness value for the set of selected sensor nodes for each setup variation and each run, in relation to the optimal value, which is calculated by a

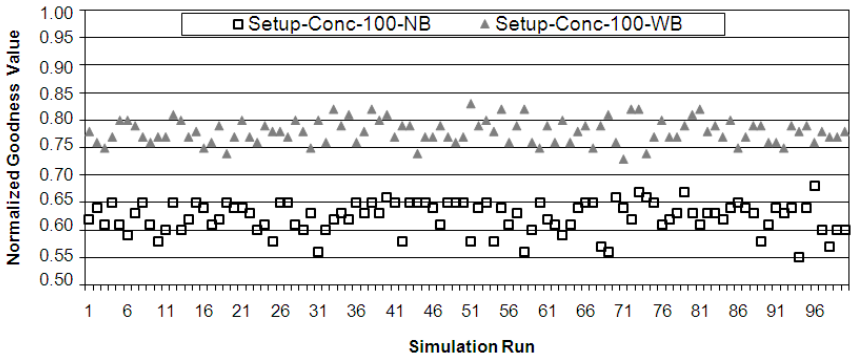
search in the global state of the simulation (in the figure, 1.00 is the normalized optimal value).

The results presented in Figures 3.16a and 3.16c show the non interference of the beeAgent in the setup variations with random node distributions for both communication ranges, as it was expected. In turn, Figures 3.16b and 3.16d present the expected better results achieved by the setups with the beeAgent for variations with node concentration, for both communication ranges. Moreover, an interesting point to be explored is the comparison between these results in relation to the communication range.

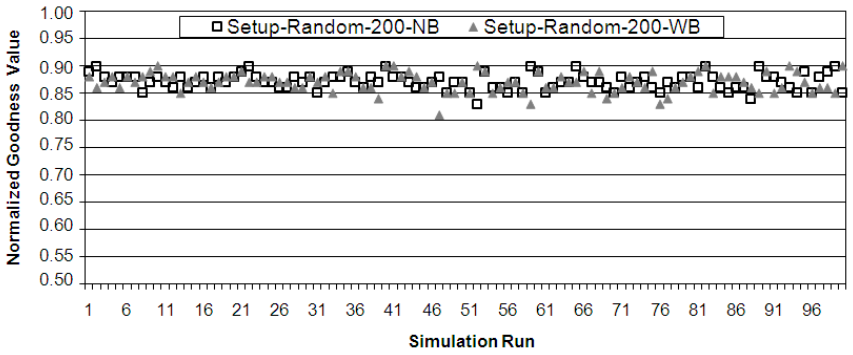
As discussed before in Section 3.3, the quality of a node's assessment in its neighbourhood depends on the number of neighbours from which it received the respective goodness values to take into account in the decision procedure. A higher communication range allows a higher number of nodes to have knowledge about more neighbours, i.e. it enlarges the nodes' neighbourhood. As a consequence, during the missionAgent dissemination, each node will receive goodness information from a higher number of other nodes, which helps to enhance its decision process. However, the difference in the results obtained with the different communication ranges shows that the gain due to the increased range is not as significant as the gain coming from the introduction of the beeAgent for the same range, in cases of the presence of regions of concentrations. Figures 3.17 and 3.18 graphically present the average of the obtained results, with error bars representing the standard deviation. In Figure 3.17 it is possible to compare setups that have regions of concentration of nodes. Observe the difference between Setup-Conc-100-NB and Setup-Conc-200-NB (0.02 units), which is smaller than the differences between Setup-Conc-100-NB and Setup-Conc-100-WB (0.157 units) and between Setup-Conc-200-NB and Setup-Conc-200-WB (0.194 units). These observations lead to the conclusion that the use of the beeAgent is more efficient than the increase of the range to handle the problem caused by the regions of node concentrations. Figure 3.18 presents the comparison among the setups with random distribution of nodes. From this figure it is possible to observe that the difference between the results from the setups with and without the beeAgent for the same range are negligible, which was expected and is coherent, as the beeAgents are not really used. On the other hand, the comparison between the setups according to the ranges reveal a difference around 0.026 units, with the better results for the setups with 200 meter range, which was also expected, as the increased range enlarges the nodes' neighbourhood, as previously discussed.



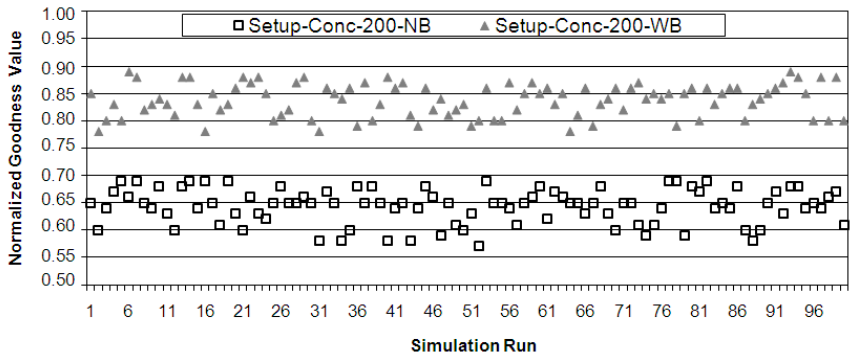
(a)



(b)



(c)



(d)

Figure 3.16: Normalized goodness values.

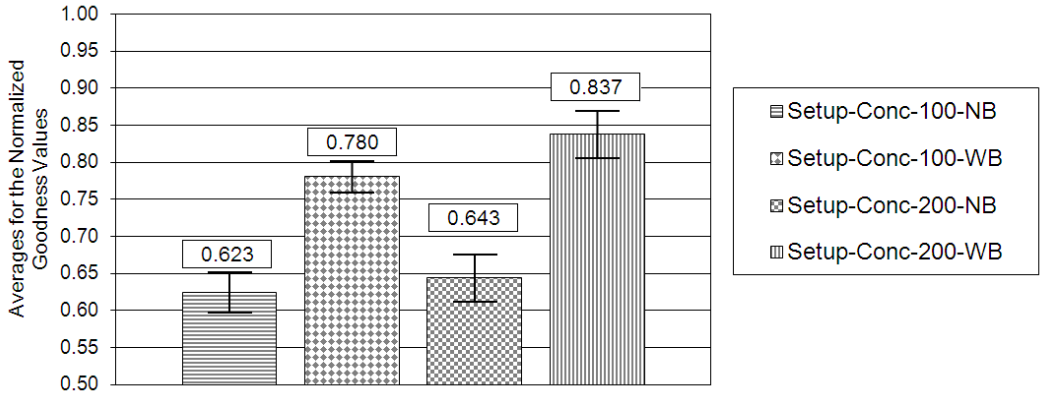


Figure 3.17: Average goodness values – Setups with node concentrations.

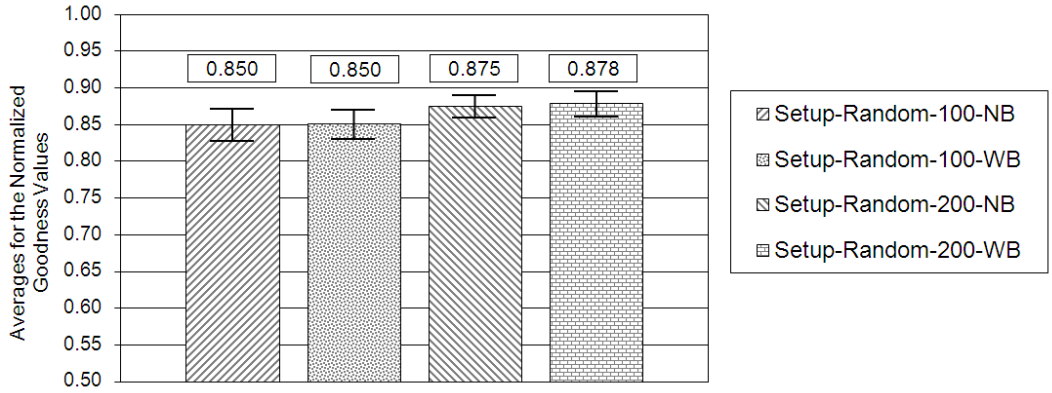


Figure 3.18: Average goodness values – Setups with random distribution of the nodes.

These findings provide evidence that the use of the beeAgents is more efficient to overcome unfavourable situations due to concentration of nodes, than just an increase of the communication range. All three metrics show that an increase in the communication range provides better results, as the goodness metric shows, while reducing the total number of messages sent. This result holds both for random nodes distributions and distributions with concentration of nodes when they are compared between them, it means the variations of Setup-Random-100-* compared to Setup-Random-200-* variations, and Setup-Conc-100-* variations compared to Setup-Conc-200-* variations. However, comparing the increased range without the use of beeAgents to the use of the beeAgents in the reduced range, for setups with concentration of nodes, namely comparing Setup-Conc-200-NB to Setup-Conc-100-WB, the use of beeAgents performs better. The metric that assesses the engaged number of nodes did not show significant influence for the different setup variations, which represents a good result, as it shows that the proposed solution, in spite of the conditions (ordinary, with a random distribution of the nodes, or adverse, with concentration of nodes), performs equally well in engaging a number of nodes close to the one required for the mission. Thus the weight of the evaluation rests on the overhead and on the goodness metrics.

Notice that these results should be carefully interpreted. Despite the benefits that an increase in the communication range presents in the achieved results, it may imply a severe augmentation in the energy required by the radio transmitter, which may not compensate the savings in the number of emitted packets and on the enhancement in the

goodness values. This issue has to be taken into account for each specific communication device used by the nodes in the WSN that is being deployed.

3.5 Summary

The proposed decentralized solution presented in this chapter explored geographical awareness to address the mission dissemination in static WSN and local context information to control sensor nodes' decisions about the mission allocation. The goal of the proposed solution was to engage a number of sensor nodes in the mission by selecting the most appropriate nodes for this particular mission, while keeping the communication overhead as low as possible. The achieved experimental results indicate that the proposed approach successfully selects high quality nodes (approximately 15% worst than the optimum solution) and in a number close to the desired one (less than 5% different from this desired number in average). Moreover, the communication overhead among the sensor nodes is expressively lower than the flooding based approach, using less than 50% of the communication used by a flooding solution in cases in which regions of concentration are detected.

4 SENSING MISSION DISSEMINATION IN MOBILE WSN

4.1 Introduction

Regarding the scenario presented in Section 1.2.2 and the related problem stated and discussed in Section 1.4.2, the goal of the solution proposed in this part of the thesis work is to assemble sensing missions and the rules telling how they move among a fleet of mobile sensor nodes to reach and stay in their mission areas (MAs). This is implemented by mobile software agents representing both the missions and their motion rules.

Considering the conditions of a mobile wireless sensor network (MWSN) as presented in Section 1.2.2, the mobile sensor nodes are not bound to the sensing missions, but they are necessary to enable the missions, provided that they can offer the conditions required to perform them. As it is assumed that every node has the same resource capabilities to perform the missions, the remaining condition to be considered is the nodes' location. Sensing missions are to be performed in MAs and as the nodes can move freely and are not bound to the missions, the missions instead have to move among the nodes to reach and utilize nodes that will carry them to their respective MAs.

In such scenario, it is expected that different users submit sensing missions to the network. As in principle there is no strong relation among the nodes and the missions, these missions can decide individually, and possibly in different ways, how they select and use sensor nodes that are suitable to help in their dissemination and execution. This autonomy aspect motivates the usage of mobile software agents to represent and encapsulate missions and the decision mechanisms needed to achieve this. To do the actual work the mobile software agents perform software migrations among the nodes. Software migrations are actions that transfer mobile software agents from a node to another node fulfilling a temporary or final goal via one or more neighbourhood migrations as presented in Section 2.2.1. Thus, the solution for the sensing mission dissemination in mobile WSN is performed by agent migrations among the sensor nodes, in which the agents have the goal to reach sensor nodes that lead them to the MAs of their corresponding missions.

Figure 4.1 shows an example in which an agent first is being carried by sensor node S-1, which is moving outside the MA where the agent should execute its mission. Besides moving outside the MA, node S-1 moves away from the MA (Figure 4.1a), i.e. in a wrong direction in relation to what that is desired. During its movement, the node S-1 however meets another node, S-2, which is moving in a direction that can lead the agent closer to MA. At this moment a decision to migrate or not to S-2 has to be taken. Assuming that the decision is positive, the agent migrates to S-2 and follows with the node in its movement. Eventually, S-2 meets another node, S-3, which is moving in the

direction of the MA (Figure 4.1b). As S-2 is not moving towards MA, the agent decides to migrate to node S-3, and finally the agent manages to arrive to its MA (Figure 4.1c).

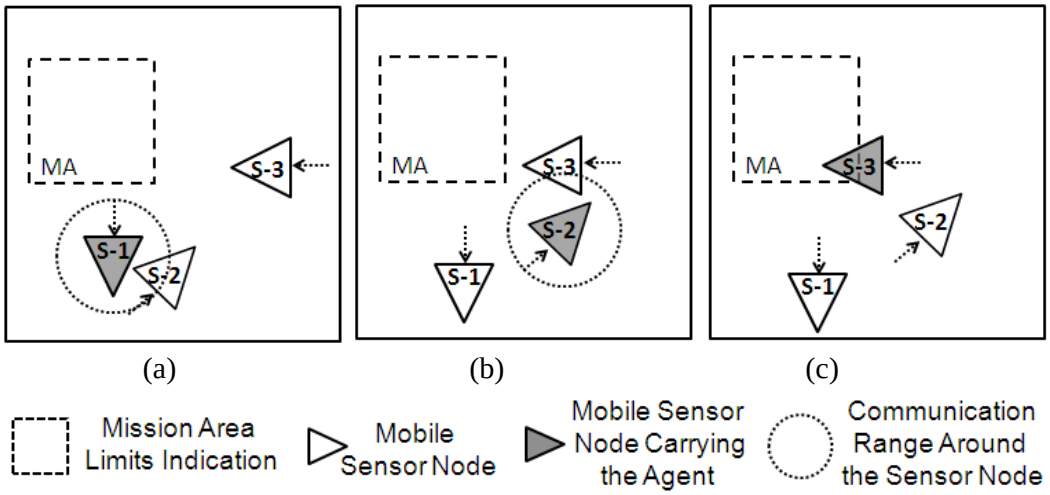


Figure 4.1: Example of a Successful Migration of an Agent Carrying a Mission.

Figure 4.1 shows an example of successful migration of an agent from a node outside the MA that was not moving towards the area (S-1) to another node that eventually carries the agent to its MA (S-3). This migration of the agent was performed by in two hops (first from S-1 to S-2 and then from S-2 to S-3), exploring the concept of agent ferrying (TEI et al., 2005) as explained in Section 2.2.1.

As presented in Figure 4.1, nodes S-1 and S-3 just meet another node during their movements and node S-2 meets two others, but in a real scenario with a higher density of nodes, node meetings happen much more often. In this case, the agents are not supposed to migrate to all nodes that they meet, but instead, they have to consider if it worth or not to migrate to a meeting node considering the costs and benefits of such an action. Thus, there is a need to have a good policy and idea about what to consider when deciding about migrating or not.

The goal in migrating software agents via mobile sensor nodes in the neighbourhood is to eventually reach nodes that are inside the MA of the respective mission by preferring nodes moving towards this area, as in the example presented in Figure 4.1. For this purpose, there is context information that can be explored to provide intelligence to the agents, so that they can take better decisions in relation to their migration actions. This context information is related to the geographic position of the nodes, as well as the directions of their next movements.

The above mentioned facts motivate the proposal and comparison of three different approaches to perform the agent migration, based on different levels of intelligence and thus processing and communication capabilities, to explore different levels of context information.

4.2 Intelligent Agent Migration

As mentioned above, the proposed approaches have in common that they explore the concept of agent ferrying to provide a solution for the sensing mission dissemination problem in mobile WSN, by means of agent migrations among mobile sensor nodes moving in the direction of the MA.

In the proposed and investigated solutions to this problem, two types of agents are used: NodeAgent and MissionAgent. These two types of agents follow the same categorisation and description as presented in Chapter 3, in which the nodeAgents are static and allocated to all sensor nodes, being responsible to provide access interfaces to the node's resources to the missionAgents, i.e., the mobile agents that handle the sensing missions.

Differently from what was presented in Chapter 3, in this chapter all nodes are mobile. Considering the node mobility, the missionAgents migrate from node to node by selecting nodes that with higher probability eventually lead them to the MA where they then can perform their missions.

The first condition that a missionAgent has to consider is if its current node is inside or outside the MA. In the first case, the agent has arrived to a node where it can perform its mission and does not need to migrate to another node. Otherwise, it has to wait for its current node to arrive to the MA to perform its mission, or try to migrate to another node, when its current one meets another node that has a better valuation. Based on these considerations, the missionAgent's behaviour, needed to analyse if it can perform the mission or try to migrate to another node, can be defined as presented in Listing 4.1.

Listing 4.1: Algorithm, in pseudo code, defining the behaviour for the missionAgent.

```
01 While (true)
02   While (Inside_MA)
03     perform(Mission);
04   End_While
05   While (!Inside_MA)
06      If Meeting_Node != null then
07          Worth ← evaluateMigration(Meeting_Node);
08          If Worth == true then
09             migrate(Meeting_Node);
10          End_If
11      End_If
12   End_While
13 End_While
```

Listing 4.1 above shows that the missionAgent is either performing its mission inside the MA (lines 02 – 04) or waiting for a meeting node while its current node is outside the MA (lines 05 – 12). When its current node meets another node, it evaluates if it is worth or not to move to this node (line 07), and if so, it migrates towards the meeting node (line 09).

The method `evaluateMigration(Meeting_Node)`(line 7 of Listing 1), is responsible for the decision about the missionAgent migration towards the meeting node. To implement this method, the missionAgent has to communicate with the nodeAgent to obtain the necessary geographical information about the current node and the meeting node so that a decision can be taken. Which information the missionAgent requests from the nodeAgent and how well it uses this information defines its level of intelligence. The nodeAgent in its turn acquire information about the meeting node by communicating with the nodeAgent in the meeting node.

This proposal considers three levels of intelligence distinguished by the information that they use to take the decision about migrating or not to a meeting node, which are detailed in the following.

4.2.1 Destination Based Reasoning

For the first level of intelligence, called Destination Based Reasoning, the missionAgent is just capable to know if its carrying node is inside or outside of the MA, and if the next destination of a node is inside the MA or not. It represents, for example, a situation in which the nodeAgent only has a course grained map and no access to additional geographic or positioning information provided e.g. by a GPS (Global Positioning System), such as the precise position or the route followed by the mobile node or its direction. Thus it cannot provide more detailed information to the missionAgent.

The `evaluateMigration(Meeting_Node)`method, implementing the Destination Based Reasoning intelligence level, performs the following sequence of steps: 1) if its current node has a destination inside the MA, the missionAgent continues in this node, i.e. it decides to not migrate; 2) if its current node does not have its destination as a position inside MA, in the case in which the destination of a meeting node is inside the MA, the agent decides to migrate to the meeting node, otherwise the agent continues in the current node. Listing 4.2 presents this algorithm, in which the evaluation of the current node's destination is presented in line 01, while the evaluation of the meeting node's destination is presented in line 05.

Listing 4.2: `evaluateMigration(Meeting_Node)`implementing Destination Based Reasoning

```
01 If (Current_Node.getDestination() == Inside_MA) then
02   decision ← false;
03 End_If
04   Else
05     If (Meeting_Node.getDestination() == Inside_MA) then
06         decision ← true;
07     End_If
08       Else
09         decision ← false;
10       End_Else
11   End_Else
12 return decision;
```

4.2.2 Direct Path Reasoning

The second higher level of intelligence is an evolution of the reasoning performed by the first. For this level, it is considered that the nodeAgent has not only the information available from the simple map, but also the exact position of its hosting node as well as the destinations of this node and of the meeting node. This is a case in which the missionAgent is assumed to receive information from the nodeAgent that has access to a positioning system such as GPS, but does not have information about routes, i.e. from a detailed map.

The reasoning mechanism used at this level is based on the probability of a node to pass through the MA. To calculate such probability, an evaluation of the direct path that connects the node's current position and the destination is done, and accordingly is called Direct Path Reasoning. The probability is computed by creating a direct path connecting the node's current position and destination position and evaluating the length of this path that stays within the MA. The greater the length of this path inside the MA for a node is, the greater the probability of the missionAgent to select this node to stay or to migrate to. If the computed probability for a meeting node is higher than for the current node, the missionAgent migrates, otherwise it stays in the current node. Figure 4.2 presents two examples of the application of this strategy.

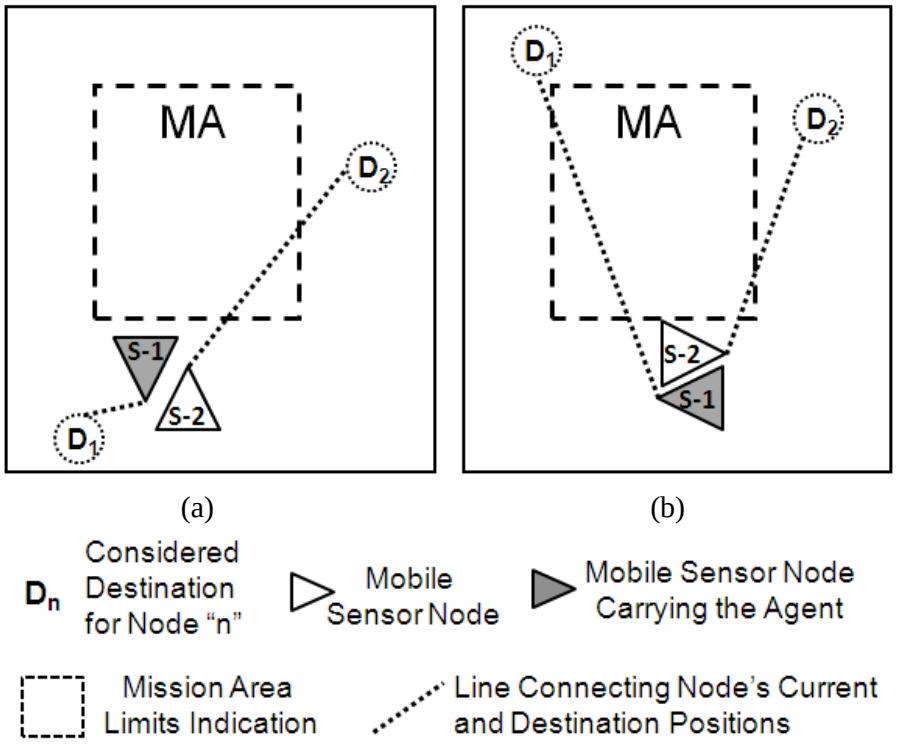


Figure 4.2: Examples for the evaluation performed by the Direct Path Reasoning.

Observe that in Figure 4.2a, the line connecting the current position of node S-1, the one currently carrying the missionAgent, does not cross the MA, while the

corresponding line for node S-2 has a portion that is inside the MA. Following the described reasoning strategy, the missionAgent will migrate from node S-1 to node S-2. In Figure 4.2b, a complementary example is presented. In this situation, the lines connecting the current positions and the destinations of both nodes have portions inside the MA, but the length of the path inside the area for node S-1 is bigger than the one for node S-2. As a result, the missionAgent that is currently in node S-1 will not migrate to node number S-2. Listing 4.3 depicts the algorithm implemented by the direct path reasoning level of the `evaluateMigration(Meeting_Node)` method.

Observe that in Figure 4.2a, the line connecting the current position of node S-1, the one currently carrying the missionAgent, does not cross the MA, while the corresponding line for node S-2 has a portion that is inside the MA. Following the described reasoning strategy, the missionAgent will migrate from node S-1 to node S-2. In Figure 4.2b, a complementary example is presented. In this situation, the lines connecting the current positions and the destinations of both nodes have portions inside the MA, but the length of the path inside the area for node S-1 is bigger than the one for node S-2. As a result, the missionAgent that is currently in node S-1 will not migrate to node number S-2. Listing 4.3 depicts the algorithm implemented by the direct path reasoning level of the `evaluateMigrate(Meeting_Node)` method.

Listing 4.3: `evaluateMigration(Meeting_Node)` implementing Direct Path Reasoning.

```
01 prob_Current ←
              Calculate_Prob(Current_Node.getDestination())
02 prob_Meeting_Node ←
      Calculate_Prob(Current_Meeting_Node.getDestination())
03 If (prob_Current > prob_Meeting_Node) then
04   decision ← false;
05 End_If
06   Else
07     decision ← true;
08   End_Else
09 return decision;
```

4.2.3 Route Aware Reasoning

The third and highest intelligence level for the missionAgent considers that the previous approaches may not present a good performance in some cases. For instance, the direct path one may fail in cases in which the node takes a route that does not match with, or deviates much from, the straight line traced between its current position and its destination point. This can be the situation in the example presented in Figure 4.2b. In this example, node S-1 may not even pass inside the MA to reach its destination, while node S-2 may take a path that effectively pass through MA. This situation is depicted in Figure 4.3. The direct path reasoning cannot consider this hypothesis, because it does not have information about the route that the nodes are going to follow.

Observing the kind of problem presented above, the third type of intelligence level, the Route Aware Reasoning, considers the complete route from the nodes' current

positions to their destinations. This feature enables the missionAgent to calculate the shortest path from the current position to the MA or even to positions closer to the MA.

The computation of the shortest path enables the agent to decide to migrate to a node that will go more directly to the MA, even in the case in which its current hosting node is also moving towards the MA. Moreover, the ability to consider destinations closer to the MA represents an improvement in relation to the other strategies. This is because for the nodes that are not moving towards the MA, but may have destinations closer to it and have no probability of passing inside the MA, the missionAgent using the route aware reasoning may consider to migrate into them, while it would not occur in the two previously presented intelligence levels. In this case, for instance, with the direct path reasoning the missionAgent would stay in its current hosting node. Conversely, for the route aware reasoning, the missionAgent would consider the complete route of each node, both the one in which the agent is currently hosted and the meeting node, in order to define which one will pass closer to the MA. Selecting the node that would pass closer to the MA increases the opportunity of the agent to meet other nodes that are moving into the MA.

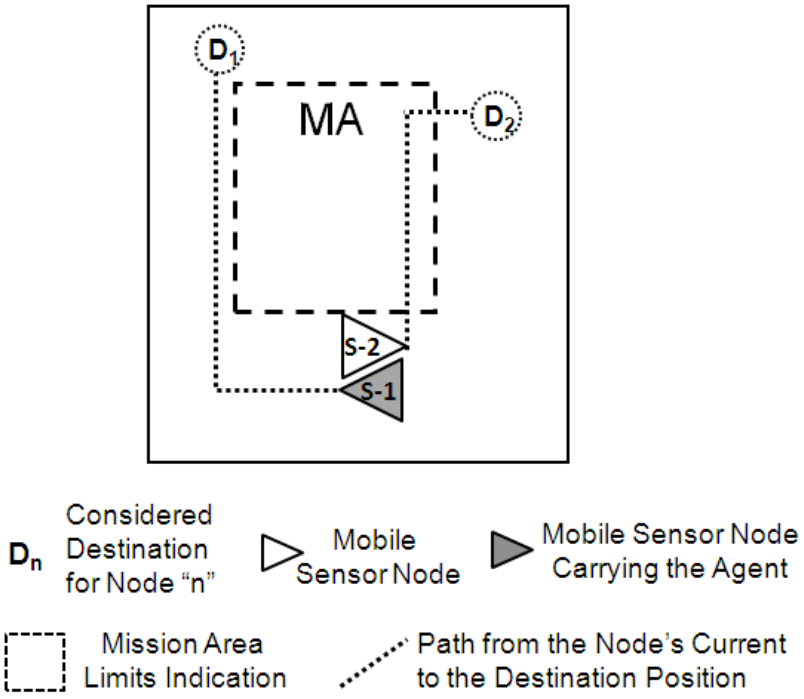


Figure 4.3: Example of complete route paths considered by the Route Aware Reasoning.

Listing 4.4 presents the algorithm used by the route aware reasoning to implement the `evaluateMigration(Meeting_Node)`method. Line 1 tests the cases in which both nodes have destinations either inside or outside MA. In this case, the decision to move or not to the meeting node considers the shortest path in relation to MA (lines 02 – 07). Otherwise, if one of the nodes has destination inside MA and the other outside MA, the decision will be to migrate if the meeting node is the one that

moves towards the MA (lines 10 – 12), or to not migrate if the current node is the one that moves towards the MA (lines 13 – 15).

Listing 4.4: `evaluateMigration(Meeting_Node)` implementing the Route Aware Reasoning.

```
01 If ((Current_Node.getDestination() == Inside_MA &&
       Meeting_Node.getDestination() == Inside_MA) ||
        (Current_Node.getDestination() != Inside_MA &&
       Meeting_Node.getDestination() != Inside_MA)) then
02         If (Current_Node.path(MA) <
                              Meeting_Node.path(MA)) then
03             decision ← false;
04         End_If
05           Else
06             decision ← true;
07           End_Else
09 End_If
09   Else
10     If (Meeting_Node.getDestination() == Inside_MA) then
11         decision ← true;
12     End_If
13       Else
14         decision ← false;
15       End_Else
16   End_Else
17 return decision;
```

4.3 Experiments and Results

4.3.1 Case Study and Simulated Environment

Simulations were performed taking a vehicular sensor network (VSN) as case study, in which the mobile sensor nodes of the VSN are a fleet of taxis. In this application, the taxis move around a city to respond requests from customers. During their movement around the city they will, with some probability, cross areas of interest of the missions, i.e. mission-areas. Therefore, the missionAgents can take advantage and ride the taxis while this is convenient for them to do so, according to the location of their MAs.

The environment used for the experiments is a squared area representing a map of a city, divided in blocks, in which a MA is defined. Figure 4.4 presents this environment which illustrates an area indicating the MA. Notice that in this “screenshot view” of the simulated environment presented in Figure 4.4, there are nodes that have a missionAgent, while others do not have it. Among the last ones, some had a missionAgent in the past, but have migrated to other nodes.

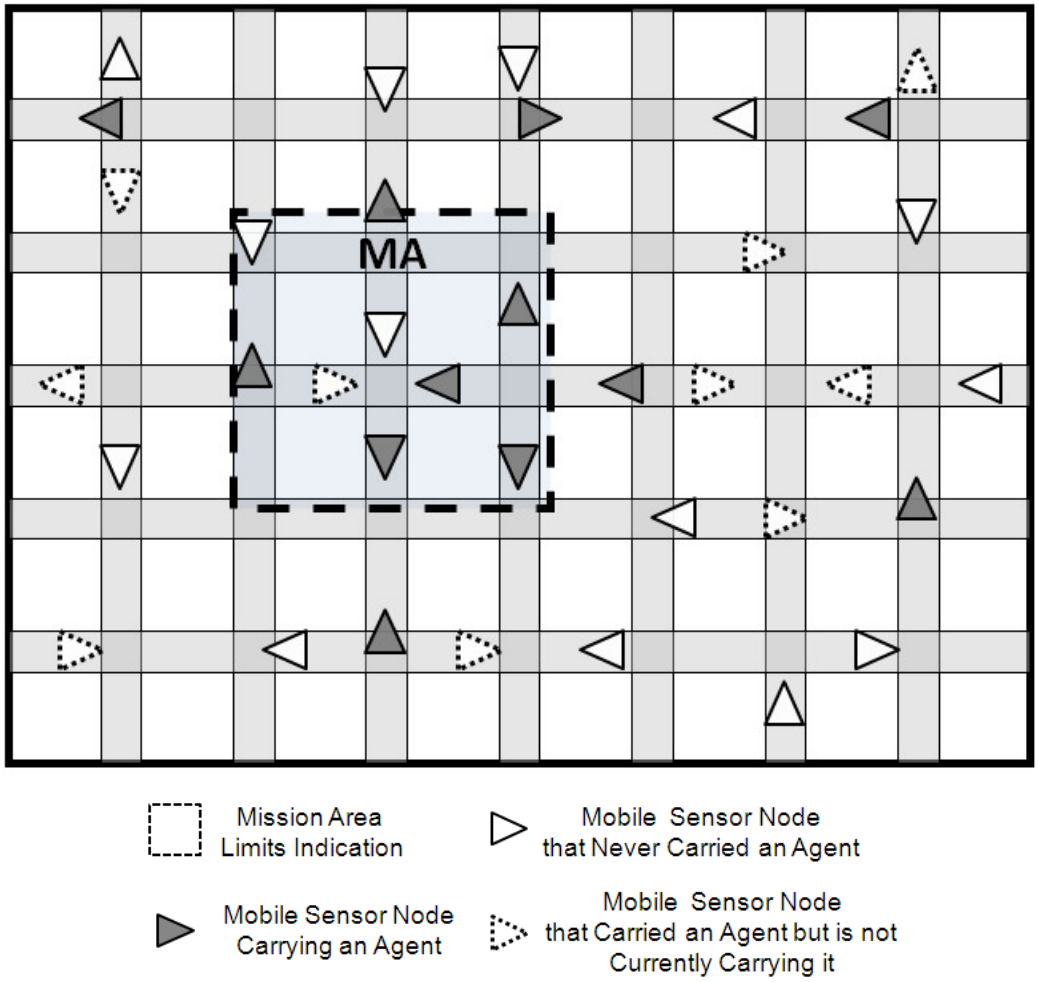


Figure 4.4: Simulated environment model.

The movement of the nodes in the presented case study is based on the Manhattan Mobility Model (BAI; SADAGOPAN; HELMY, 2003) and modelled according to the following. The nodes move along the streets between the blocks that compose the city scenario. They select a given point of one street and move towards it. When a node reaches an intersection, it chooses one direction to follow: north, south, west, or east. This choice of direction is random but considers the direction that the node is driving to, i.e. if the node is moving to a point located at north-east of its current position, it randomly select to move north or east.

If the node is a taxi with a defined destination, it means, a taxi that is carrying a passenger or is going to take a passenger, it moves preferably in a straight line towards the destination point. This is done by assigning a higher probability to the choice of moving forward, it means to continue in the same direction, or to turn into the direction of the destination. For instance, if the destination is in a north-east location in relation to the current node's location and the node is facing north, it moves preferably straight to the north direction, until reaches the north level of the point. Then it turns east and

follows this direction until reaches the destination. When a node reaches its destination, it starts to search for a passenger or goes to another destination, i.e. takes another passenger.

A taxi that is driving without a specific destination, i.e. searching for passengers, is given equal probability to move forward as to turn left or right, and a lower probability to move backwards. The backward movement has a lower probability in order to avoid unrealistic “back-and-forth” movements. In this state, when a taxi driver is searching for passengers, the destination considered by the agent reasoning is the next street intersection, which is the decision point in which the node will decide where to go next. Then, the nodes either may be taken by a new passenger, thus acquiring a destination to move to, or continue the search. If it gets a new passenger it will move towards its new destination according to the movement pattern described above.

4.3.2 Simulation Setup

The simulated environment is a 1.8 Km $\times$ 1.8 Km area divided in 20 $\times$ 20 blocks of 90 meters side each. The considered MA has dimensions of 4 $\times$ 4 blocks. The sizes of the entire environment and the MA are adequate to the evaluation of the proposed approach, as the region around the MA represents the surroundings where the agents should find a node to move to and come back to the MA, otherwise they will stay to far from it.

The number of simulation runs was set to 100, each representing 30 minutes run. The nodes moved in the scenario with speeds varying between 10 and 50 Km/h, a realistic range for cars driving in urban areas of a city. Two variations in the numbers of nodes that populate the scenario were tested, namely 30 and 60, and 5 agents were created to migrate around them. The number of agents was empirically defined due to the fact that values lower than 5 provided very poor results for the Random Reference used for comparison purposes. In the beginning of the simulations, these agents are deployed in 5 randomly selected nodes. The communication among nodes is performed with omni-directional propagation model with a range of 90 meters, which is fairly realistic even considering an environment such as a city in which the blocks with buildings hinder the wireless communications, as discussed in (GIORDANO, 2010). The wake-up period to search for neighbours to communicate with (meeting nodes) was set to 5 seconds. This value was empirically established. It was verified by simulations that values smaller than this did not provide any significant improvements in the main results. On the other hand, values greater than this one negatively impacted the results, as the nodes may have a significant displacement depending on their actual speed, then loosing the opportunity to communicate with meeting nodes. A summary of the simulation parameters is presented in Table 4.1.

Table 4.1: Simulation parameters.

Parameter	Value
Area Dimensions	1.8 Km $\times$ 1.8 Km
Block Dimensions	90 m $\times$ 90 m
MA Dimensions	4 $\times$ 4 Blocks
Simulation Time	30 minutes
Nodes' Speed	10 Km/h – 50 Km/h
Number of Nodes (Density)	30 (9.25 nodes/Km^2) 60 (18.51 nodes/Km^2)
Number of Agents	5
Nodes' Communication Range	90m
Broadcast period	5 seconds

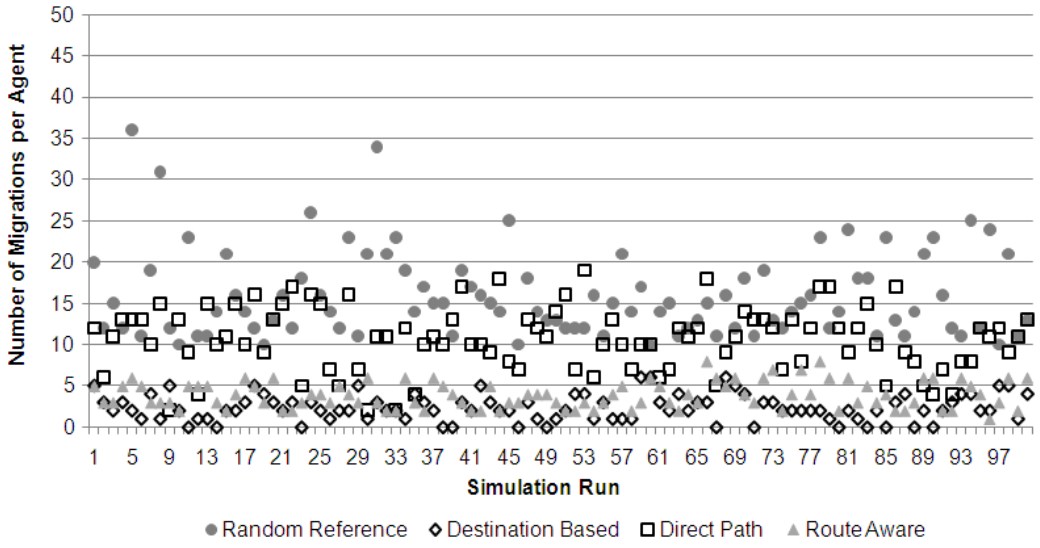
Besides the three intelligence levels presented above, an additional one was also simulated to be used as reference for comparisons. This type represented a "dummy" agent which performed a random decision to migrate or not when its current node meets another.

4.3.3 Results and Discussion

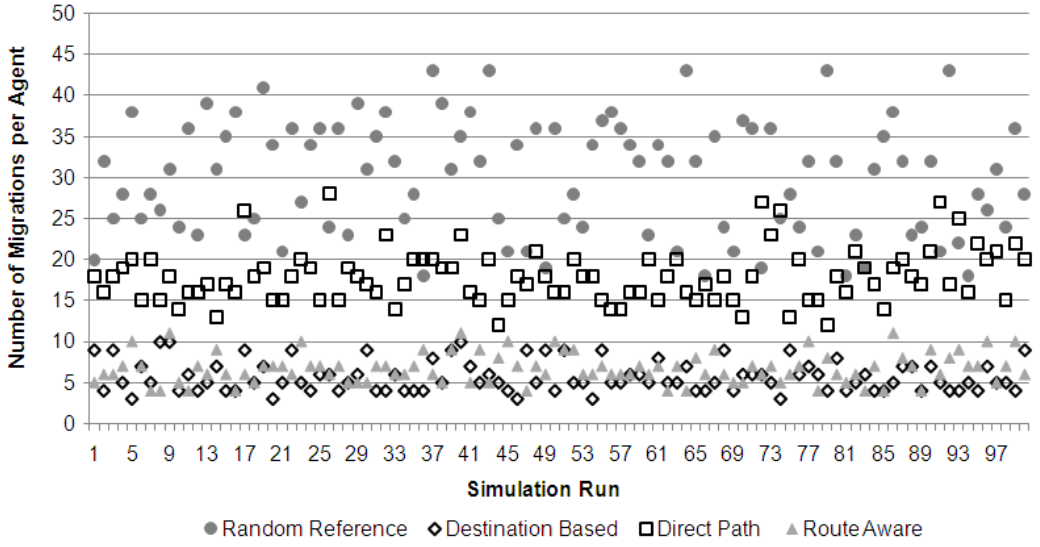
The evaluation of the different levels of intelligence presented above was done by means of two metrics: 1) Number of performed migrations per agent in each simulation run; and 2) Percentage of the simulation time during which the agents were inside the MA. The first metric provides an insight about the overhead in terms of usage of communication resources. The second provides information on how efficient each model of intelligence is in driving the agents towards and keeping them inside the MA. It is important to notice that the first metric is considering the overhead due only to the migration of the agent itself, and this is why the results are presented in terms of number of migrations instead of transmitted bytes per migration. This is explained by the fact that in this work it is considered that the data (or state) transmitted with the missionAgents during their migrations are of similar size for all agents, which is a reasonable assumption, considering that in a real usage of this approach, they would only carry data processed by algorithms that they would implement as services, e.g. data aggregation, and would not carry large amounts of raw data.

Figure 4.5 presents the results for the first metric, grouped according to the two different simulation variations in relation to the number of nodes, namely 30 nodes (Figure 4.5a) and 60 nodes (Figure 4.5b). The results are presented for an average of

migrations that the 5 agents do in each simulation run according to their different intelligent levels (Random Reference, Destination Based, Direct Path or Route Aware).



(a)



(b)

Figure 4.5: Number of migrations per agent: (a) Simulations with 30 nodes; (b) Simulations with 60 nodes.

The results reveal that two levels (the Destination Based and the Route Aware ones) presented a lower numbers of migrations, while the Direct Path one has a higher number of migrations in relation to these two first in both setup variations with 30 and 60 nodes. However, it is remarkable the difference of the Random reasoning level, which in both setups presented much more migrations than the other intelligent solutions. These

observations can be understood by the way each level performs the reasoning. Both the Destination Based and the Route Aware levels only decide to migrate when they are "sure" about the value in migrating to another node, by analysing the information that they are capable to analyse. On the other hand, the Direct Path one "risks" more. For instance, if the missionAgent using the Direct Path reasoning is in a node that has 50% of chance to pass through the MA and it meets another node with 51%, it migrates to this node. On the other hand, the Random one does not take any information in consideration, and just randomly decides to migrate or not. Thus, as the decision is based on a uniform random distribution of a Boolean variable, it is possible to state that each time it meets another node it has 50% of chance to migrate towards this node or stay in its current one. As the average number of meetings for the simulations with 30 nodes was 33.66 and for the variation with 60 nodes the average was 62.55, the average number of migrations executed with the Random reasoning is consistent, respectively 15.92 and 29.94 in each variation.

Analysing the averages along the simulation runs, the Destination Based reasoning presents the lower numbers of migrations, followed by the Route Aware and the Direct Path one. As mentioned before, the Random provides the worst result. Figure 4.6 presents these averages with error bars representing 95% confidence interval.

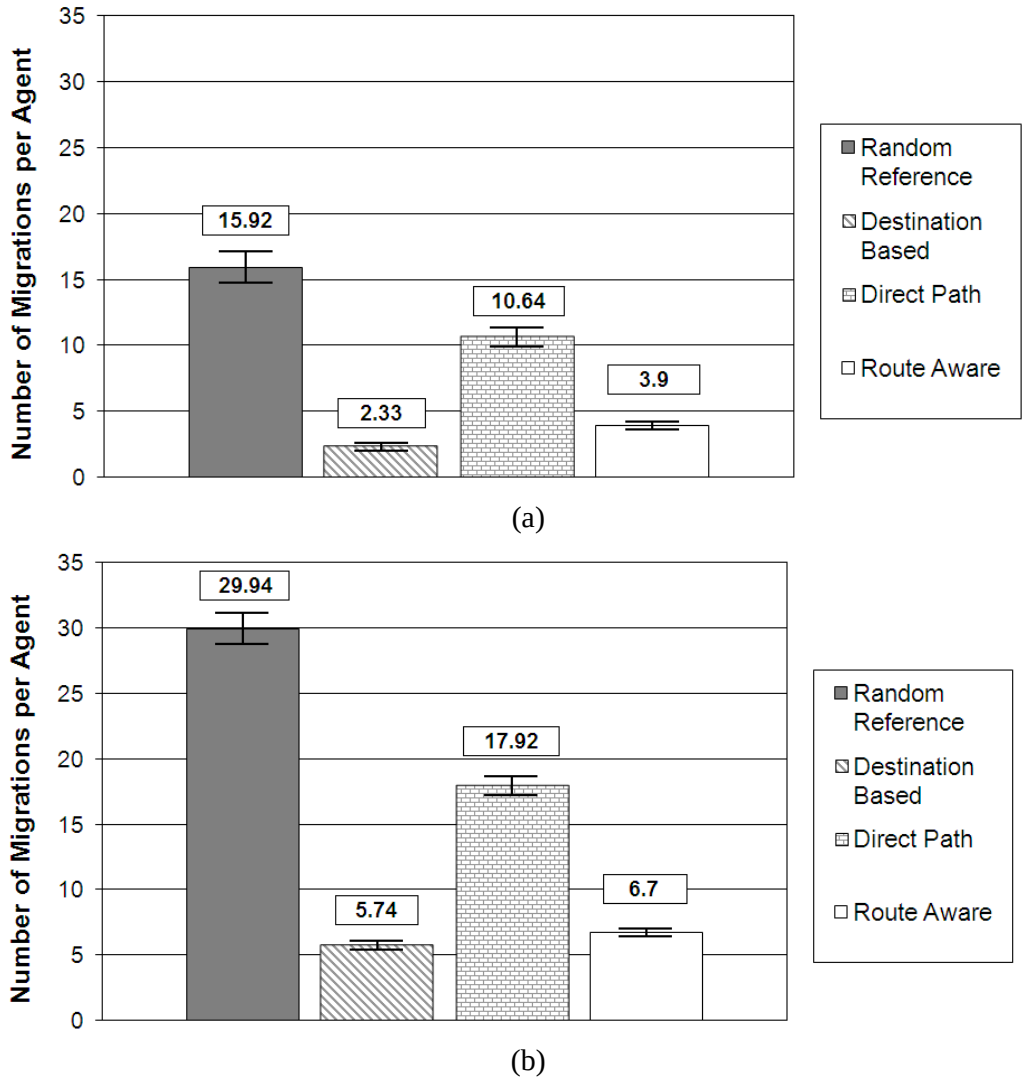
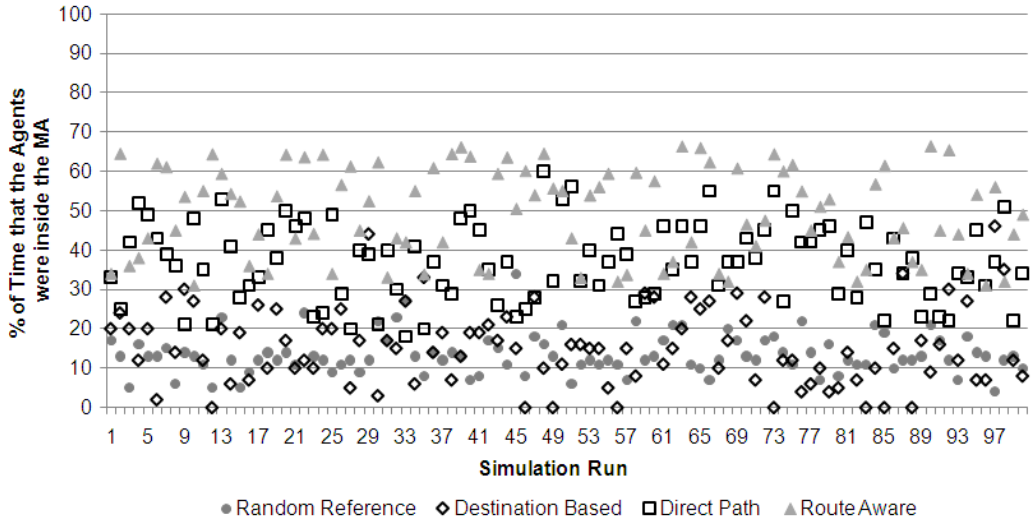


Figure 4.6: Average Number of migrations per agent: (a) Simulations with 30 nodes; (b) Simulations with 60 nodes.

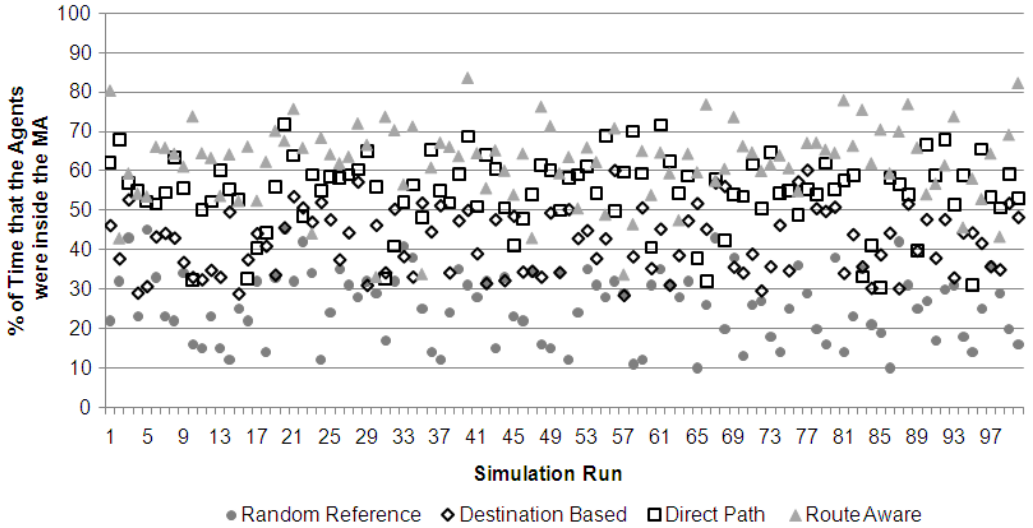
Figure 4.7 provides results for the second metric that evaluates how efficient each reasoning level is in keeping the missionAgents in the MA during the simulations. Following the same presentation as in Figure 4.5, the values for each run are displayed in the plots. The graphs plotted in Figure 4.7 are a bit intertwined, which is explained by the high variation of the acquired results, as summarised in Table 4.2, which presents the average and the standard deviation for this second metric.

Despite the high variation of the results, it is possible to observe that the Random reasoning clearly presents the lowest values in most of the runs for both setups with 30 and 60 nodes. The other three types of reasoning provide better results in average, keeping the agents in the MA. Clearly the Direct Path and the Route Aware present better results in the first simulation set with 30 nodes (Figure 4.7a). The difference between their results in relation to those achieve by the other two agents, Random and the Destination Based, is diminished in the second variation, with 60 nodes, in which

these last two obtained better results. This indicates that the Destination Based and the Random agents are more sensitive to the node density if compared to the other two more intelligent ones. This observation becomes clearer by observing the average of the simulation runs for each type of agent presented in Table 4.2 and graphically in Figure 4.8, in which the averages are presented with error bars representing 95% confidence interval.



(a)



(b)

Figure 4.7: Efficiency in terms of percentage of the simulation time that the missionAgents spent inside MA: (a) Simulations with 30 nodes; (b) Simulations with 60 nodes.

Table 4.2: Average and standard deviation values for the metric that assesses the percentage of the simulation time the missionAgents spent inside the MA.

Number of Nodes	Intelligence Level	Average	Standard Deviation
30	Random Reference	13.53	5.09
	Destination Based	15.64	9.85
	Direct Path	36.76	9.86
	Route Aware	49.25	11.36
60	Random Reference	26.02	8.99
	Destination Based	42.06	8.08
	Direct Path	54.32	9.37
	Route Aware	62.21	9.93

The cross-analysis of the results of both metrics makes possible to consider using context awareness to provide support to applications running on top of mobile nodes; such as the one proposed in this thesis. Even the Destination Based reasoning level, provides fairly good results, which are improved by adding the capability to analyze more information, i.e. the “upgrade” to the Direct Path, and finally these results present even more improvement with the upgrade represented by the Route Aware level. Despite the drawback presented by the higher overhead of the Direct Path reasoning in relation to the Destination Based, the better results achieved by the Route Aware in both metrics show the value in using more context information compared to the use of less (as the Destination Based reasoning does) or no context information at all (as the Random Reference approach).

It is noteworthy to mention the improvements in the results achieved by these two levels, the Destination Based and the Random ones, when the node density is higher. Indeed, with more nodes deployed in the area, these two agents managed to perform much better then in the simulations with less nodes. This is especially true for the case of the Destination Based reasoning level, which managed a result almost three times better in simulation with higher number of nodes than the one achieved in the variation with lower number of nodes, while still doing fewer migrations. This result is evidence that even the simpler approaches are useful in more dense populated environments. However, it is important to highlight that the efficiency of the lower level of intelligence based approaches depend on the node density, achieving better results only in the high density scenarios, while the higher levels of intelligence performed well in both cases with low and high density of nodes.

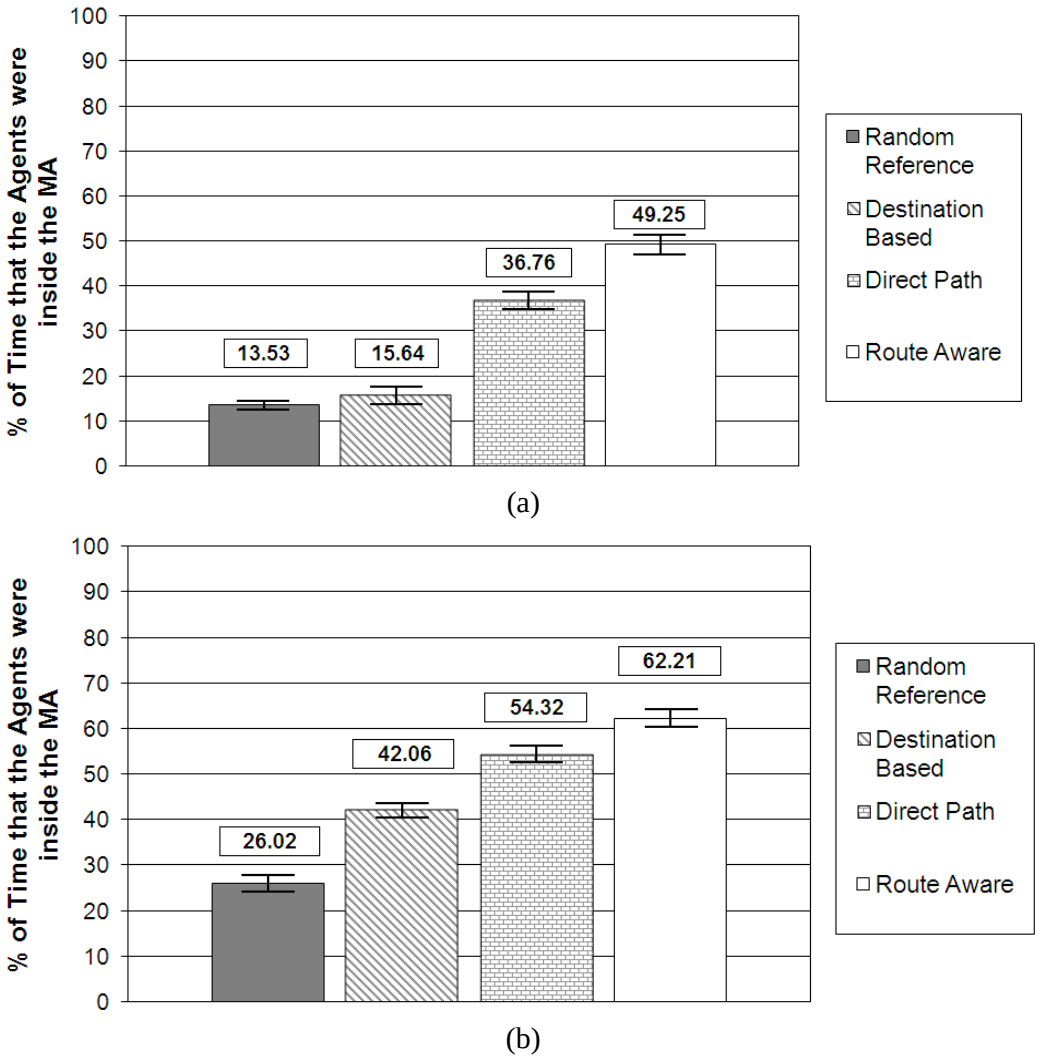


Figure 4.8: Average Number for the percentage of the simulation time that the missionAgents spent inside the MA: (a) Simulations with 30 nodes; (b) Simulations with 60 nodes.

4.4 Summary

Aiming to solve the problem of mission dissemination in mobile WSN, this chapter presented an approach in which mobile software agents that disseminate and execute sensing missions use geographic information to support their decisions. Different levels of intelligent decisions based on three distinct levels of information completeness are proposed and compared. The experimental results indicate that the increased information richness provides better results in keeping the agents responsible for a mission in their mission area, which is the main goal in the performed experiments. However, the associated cost due to communication does not follow the same trend, as it can be noticed by comparing the results achieved by the Direct Path approach to those achieved by the Destination Based. Despite of maintaining the agents more time inside

the mission area, the Direct Path approach requires approximately three times more communication among the nodes than the Destination Based one. The Route Aware approach, which is the one that uses the most complete information, achieves the best results keeping the agents inside the mission area for more time, and having a low cost associated with the nodes' communication. This low cost due to communication is very similar to the one achieved by the Destination Based approach. Nevertheless, the hypothesis that it is worthy to use geographical context for the mission dissemination in mobile WSN is confirmed by the comparison of the results achieved by the three proposed approaches to those achieved by a random reference solution. The random solution requires more communication (almost 50% more than the Direct Path in the case of high node density), and presents poor results in relation to keeping the agents inside the mission area (approximately 61% of the time that the Destination Based approach manage in the case of high node density).

5 ALARM HANDLING IN COOPERATIVE STATIC AND MOBILE WSN

5.1 Introduction

Observing the scenario presented in Section 1.2.3, the combined use of static and mobile sensor nodes for surveillance applications is a promising approach. The possibilities of such a combination are even larger if also considering mobile sensors in air, as those addressed in the presented scenario. There are different arguments that can be used to advocate this statement, but the improved coverage achieved due to increased mobility summarise most of them (GIAMBERARDINO; GABRIELE, 2008). However, as described in Section 1.4.3, the combination of these two types of sensor nodes offer challenges that have to be tackled so that the overall system can efficiently respond the final users' expectations, avoiding waste of resources.

As intrinsically distributed, a surveillance system as depicted in the presented scenario would hardly scale if it is dependent of a single central entity to organize the interactions among the nodes that compose the system. The main reasons for that are related to the size of the network, the non-determinism of the events that it has to handle, as well as its desired responsiveness to these events.

In respect to the size of the network, this kind of system is expected to employ a large number of static nodes and smaller number of mobile nodes, where the specific quantity depends on the available budget and user requirements. Thus, a central entity collecting information from all these nodes and issuing specific orders about what they should do next is clearly undesirable because it would imply an enormous traffic of control messages. This overuse of communication to transmit control messages would represent a waste of resources, regarding that communication is an expensive operation in terms of energy consumption, as already discussed. Moreover, a central information collector and order issuer would represent a single point of failure, which is also an undesirable characteristic.

Regarding the events that such a network has to handle, they are expected to happen in any location of the network, and in principle, events happening in a given location have nothing to do with events happening at other locations. This locality characteristic demands local interactions among the closest nodes in order to engage appropriate resources close to the place where these events occur. However, it is possible that this interaction may trigger the request for remote nodes, but in principle the engagement

and preference of resources selected from the close by neighbour nodes is a desirable characteristic.

The system responsiveness is also related to the above two mentioned reasons. This connection is explained by the fact that to develop a surveillance system that can effectively and timely respond incoming demands, a central entity to process them would represent an inacceptable bottleneck, while the scattered pattern of events demands local and concurrent handling.

In order to handle the complexity described above, the network design has to take into account aspects that the nodes should present to successfully address the problems that emerge from their complex interaction. These aspects can be summarised as follows (DRESSLER, 2007):

a) Autonomous behaviour control;
b) Loose coupling;
c) No need of a global state;
d) No (global) synchronization;
e) Dependence on the environment;
f) Possibly cluster-based collaboration.

Observing these aspects, self-organizing solutions for massively distributed system, such as the surveillance systems composed of static and mobile sensor nodes addressed in this thesis work, present a plausible approach (DRESSLER, 2007). According to this statement, involved nodes should present autonomous decentralized behaviours, being loosely coupled, but also being able to locally collaborate with peer nodes in groups or clusters, without requiring global synchronization via central entities. As there should be no central entity controlling all the nodes, the system should not present any global state either.

In light of the problems presented in Section 1.4.3, and the above mentioned observations, this chapter presents a contribution in providing a cooperative use of static and mobile sensor nodes to be used in surveillance systems. To provide a self-organizing alternative for the interaction among static and mobile sensor nodes, this chapter presents and shows the benefits of an approach inspired by the biological process and behaviour of ants constructing and following trails to locate food.

To support the above goal, first the adopted assumptions and definitions of the elements composing the application scenario are presented. In the following it is presented the proposed technique based on the ant pheromone trail analogy to tackle the identified alarm delivery and assignment problems. Finally, extensive experimental results are presented along with their related analysis.

5.2 Definitions and Assumptions

Considering the scenario presented in Section 1.2.3, the proposed approach to address the alarm delivery and assignment problem is based on a multi-agent solution in which each static and mobile sensor node, is handled by a software agent (a nodeAgent as in Chapters 3 and 4), which provides the sensor nodes' with intelligence. Thus, every action made by a sensor node is in fact decided by its respective nodeAgent. By means

of interactions among nodeAgents, sensor nodes cooperate to handle threats in the area under surveillance. To perform this handling, first the detection of possible threats is performed by static sensor nodes placed on the ground, followed by a threat confirmation performed by a mobile sensor node (carried by a UAV). Once a UAV is called by the static sensor node that detected the threat, it moves to the position of this node to confirm the threat.

To carry out this detection, search and confirmation process in which the static nodes call the UAV carried sensors, a bio-inspired idea is proposed. In this proposal, the UAVs act like ants that leave pheromones on the environment to form trails that can be followed. These trails can be of different flavours, in the case in which there are differences among the sensors carried by the UAVs. As tightly connected to the environment where they are deployed, the static sensor nodes represent the environment where the pheromones left by passing UAVs are deposited. As discussed in the scenario presentation (see Section 1.2.3), the static sensor nodes emit alarm messages to call the UAV carried sensors to handle a given threat. The idea is to make these alarms behave like ants that search and follow the trails left by other ants, i.e. the UAV carried sensors. To carry out this proposed solution, the alarms are represented as mobile software agents sent by the nodeAgents of the nodes that detect a threat. This type of mobile agent is called ***alarmAgent***.

In the case in which the pheromones left by the UAVs have different flavours, the alarmAgents, behaving like ants, are able to recognize the different flavours of these pheromones. Having this ability, the alarmAgents are able to select the trails of those UAVs that carry the most appropriate sensors to handle the confirmation of the threat announced by their respective alarms.

The different elements that compose the application scenario are presented in the following.

Mission Area

The considered scenario is composed of a mission area in which each element (threats and sensors) is identified by its Cartesian coordinates, x and y. This area may be subdivided in sub-areas, consisting of a sub-set of positions in the area, which may have properties assigned to them, such as weather conditions for a specific sub-area. These conditions can be for instance incidence of fog, mist or any type of condition that may interfere in the sensors measurements.

Threat

A given threat τ_i^k is of kind $\boldsymbol{k}$ and has an identifier $\boldsymbol{i}$, which represents the order of its occurrence in the mission area. The threat occurrence means its detection by a sensor node. There are $\boldsymbol{K}$ possible types of threats, so $\boldsymbol{k} = 1,\ldots, \boldsymbol{K}$. The threats may be static or mobile, depending on the application semantics. If mobile, they are considered to move with a constant speed $\boldsymbol{v_{\tau i}}$, but different threats may move with different speed. Mobile threats may also randomly change their movement direction.

The appearance of threats is defined by a given probabilistic arrival model, such as a Poisson distribution in which a factor λ determines the number of appearances during a

given time interval, or by a deterministic model that describes specific conditions of threats appearance.

Static Sensor Node

The static sensor nodes are identified by their corresponding coordinates (***p=(x,y)***), and this position is assumed to have been established during system deployment and does not change during system operation. It is also assumed that static sensor nodes know their own position, which is possible by means of a GPS device (PARKINSON; SPILKER, 1996) or any other positioning mechanism, such as algorithmic solutions (DOHERTY; GHAOUI; PISTER, 2001) (NICULESCU; NATH, 2003a), or their positions can be stored by an external agent when static sensor nodes are deployed.

These nodes have their communication capabilities defined by a communication range (***r_c***).Their sensing capabilities are defined by a sensing range (***r_s***) and the types of threats they may identify are represented by a set of values from ***k***.

The static ground sensor nodes behaviour is to be realistic modelled, assuming the influence of an energy saving mechanism (LIN; HE; XIONG, 2006) (YUE; SUN; JIN, 2010), in which the nodes sleep most of the time, which means that they turn off all or almost all their devices. A duty cycle mechanism defines the periods in which the nodes wake up (***t_w***), i.e. they turn on their processing, communication and sensor devices to process information, exchange messages with other nodes and sample the environment. Figure 5.1 shows a finite state machine (FSM) model that represent the ground sensor nodes' behaviour, with the two possible states, active and inactive (sleep), and the transitions between them.

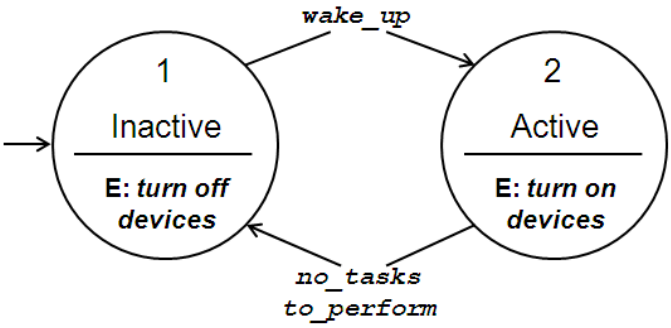


Figure 5.1: FSM for ground sensor nodes.

As an essential part of their behaviour these sensor nodes are supposed to be configured to emit alarms, i.e. the alarmAgents mentioned above, when they detect a possible threat. This important part of their behaviour is assumed to be setup by a mission dissemination and allocation method as presented in Chapter 3. In this configuration, the additional fields seven and eight of the mission structure (see Figure 2 of Chapter 3) are used to respectively define the threat detection algorithm and the alarm emission action, to be triggered when a potential threat has been detected.

Mobile Sensor Node (UAV)

The UAV instance i (denoted u_i) is considered to have an internal state $S_{ui}(t)$ at a given time t, which contains two components: a physical state and an engagement state.

a) Physical State: this state includes information about u_i's current position $p_{ui}(t)=(x_{ui}(t),y_{ui}(t))$, speed $v_{ui}(t)$, heading angle ($\psi_{ui}(t)$), communication range (r_c), sensor type (j), sensor status ($\varphi^j_{ui}(t)$) and energy resources ($e_{ui}(t)$).

b) Engagement State (ES): according to the detected threats in the surveillance area and to the respective alarms issued, a UAV can be in one of the following states: ***idle***, ***engaged***, ***negotiating*** or ***busy***. The first state may occur when a UAV is just patrolling the area, being considered idle and ready to engage in performing a task over a threat informed about by an alarmAgent. The second state occurs when a UAV is engaged in performing a task related to a threat, but it is not performing it yet, e.g. it is still moving towards the location where the threat was detected. The third state occurs when a UAV negotiates a given alarm with another UAV. The forth state, finally, occurs when a UAV is handling a given threat, i.e. performing a task over it, for example to confirm the threat. The set of states is represented by:

$$ES = \{idle, engaged, negotiating, busy\}. \quad (5.1)$$

Received alarms are stored in a queue and are handled according to a first come first served policy.

Figure 5.2 presents the FSM for the UAVs, in which it is possible to observe the possible states and transitions among them. The transition from state 1 (Idle) to state 2 (Engaged) happens when a UAV receives an alarm delivered by an alarmAgent and assumes the responsibility to handle it. If a UAV is responsible for handling an alarm and it meets another UAV, it switches from state 2 (Engaged) to state 3 (Negotiating). In this state the UAVs decide which of them has the best conditions to respond an alarm, e.g. the one that has the most suitable sensor to handle a threat, and should take the responsibility for a given alarm. If the result of a negotiation is that a UAV should hand it over to another, it releases itself from the responsibility of that alarm and if it is responsible by other alarms, it transits back to state 2 (Engaged), otherwise it transits to state 1 (Idle). If an idle UAV meets another that has alarm(s) to negotiate, it transits to the Negotiating state and if it is decided that it should assume the alarm it transits to the Engaged state, otherwise it comes back to the Idle state. Once an engaged UAV reaches the threat location, informed about in an alarm, it transits to state 4 (Busy). When it finishes handling the threat it transits to the Idle state, unless it is engaged also with any other alarm and transits back to Engaged.

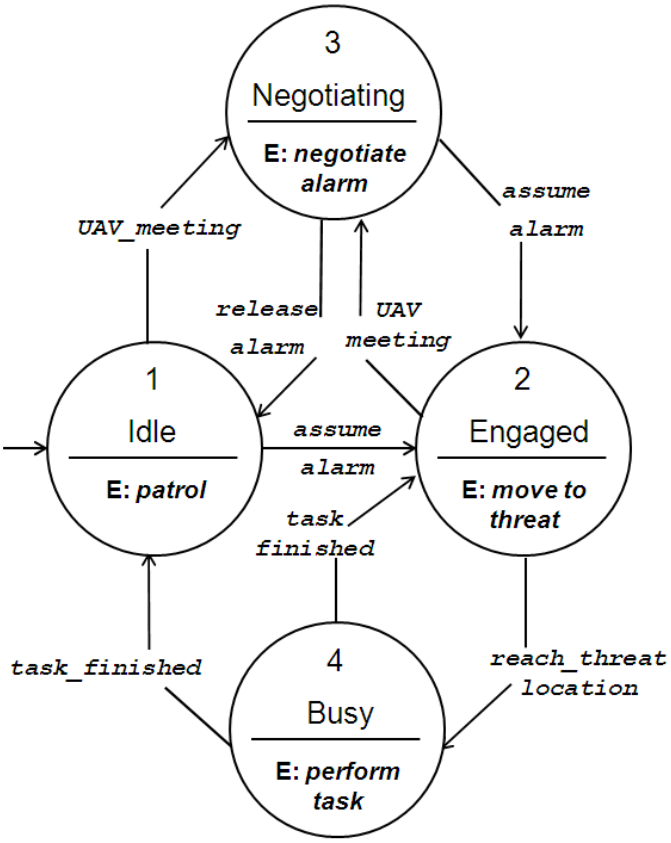


Figure 5.2: FSM for UAVs.

The adopted kinematic model considers that the UAVs move along continuous trajectories with constant speed and with a constrained turning angle (JIN et al., 2006). An additional assumption is added to the model presented in this work, allowing the UAVs' maximum speed to be higher than the maximum speed of mobile threats. This assumption allows the system to have a high-level of responsiveness to handle new threats.

The sensors that equip the UAVs are able to detect members of a set of possible types of threats and then perform more advanced tasks such as analyze or track a selected subset of these types of threats. In the case that a sensor, needed for analysis or tracking, is missing or does not match with the type of the threat, poor results are expected. The range of the detection as well as the analysis and tracking capabilities are tunable, according to the types of sensors that equip the UAVs in the fleet. This is done by adjusting the radius around the UAV that controls the range of the surveillance area in which it is able to detect and/or analyze/track a threat, identified as r_s (sensing range). A UAV changes from the state engaged to busy when the position of the threat that it is supposed to handle is covered by its sensing range, and it remains in this state while it is handling the threat.

Based on information about the type of threat, the sensor that equips the UAV (type and status) and other operation and/or environmental conditions, such as weather

conditions or remaining available energy for instance; it is possible to determine the feasibility to perform a given task related to a given threat with that sensor. This is expressed by θ, the applicability of a sensor that equips a UAV to handle a threat, which is proportional to the sensor (***j***) capability to perform a given task over a certain type of threat (***k***) at a specific time instant (***t***) in a given position ***p=(x,y)***.

$$\theta_{i,j,p}(t) = \begin{cases} u_i^j(t) \cdot Op_{j,p}(t), & \text{if } k \in K_j \subset K \\ 0 & \text{otherwise} \end{cases} . \quad (5.2)$$

where $Op_{j,p_\tau}(t)$ is a function that estimates the degradation in the measurements offered by a sensor of type (***j***) due to the operation conditions at time (***t***), which may possibly be dependent of the position $p_\tau=(x,y)$ where the threat was detected, and K_j is the subset of all types of threats containing those that match the sensor type (***j***). Poor or not at all useful results offered by sensors that do not match or are not suitable for the type of threat are mapped to a value equal to zero.

5.3 Pheromone-based Alarm Delivery Concept

To address the alarm delivery problem, the proposed approach uses a decentralized mechanism, with artificial pheromones, inspired by the biological mechanism used by ants to track food in the nature (DORIGO; DI, 1999).

Artificial pheromones are usually applied to distributed coordination by means of stigmergy, the indirect communication using environment cues (BONABEAU; DORIGO; THERAULAZ, 1999). Pheromone marks are deposited in the environment forming a trail while biological entities such as ants are moving. The pheromone provides information to other entities when they pass over it. Artificial pheromones also loose their strength over time, modelling the evaporation of the real pheromones.

In this work, the pheromones are used to guide the alarmAgents issued by a ground sensor nodes throughout the network until the alarm is delivered to a UAV that carries a sensor able to handle it. When a threat is detected and an alarmAgent is issued, the alarmAgent is responsible for locating a UAV to respond to the alarm. This is performed by routing the alarmAgent to the UAV that has the strongest pheromone marks over the area. Then, the alarmAgent delivers the alarm to the UAV, which will move to the area where the alarm was generated. This strategy is denominated ***heuristic-P***.

Following the above outlined principles, UAVs that are not handling any threat (ES equal to idle or engaged) leave pheromone marks over the sensors on the area which they cross, by means of broadcasted beacon messages. These pheromone marks are collected by the ground sensor nodes that are deployed in the area through which the UAVs have passed. When a threat is detected by a ground sensor node, it issues an alarm that is routed through the network by the alarmAgent, as already mentioned. The alarm delivery will be performed by the alarmAgent through a routing mechanism in which the alarmAgent acts like an ant that moves in the direction that points to the UAV that has the strongest pheromone marks in that area. This means that the alarmAgent

migrates among the nodes through a pheromone trail in the direction that points to the UAVs that most recently passed that location. Heuristic-P is inspired in (HEIMFARTH; JANACIK, 2008), which presents a pheromone-based strategy to migrate services in a sensor network. In this referred work, the pheromone concentration determines the places where the services are required. In heuristic-P, instead of services, alarms move through the network, by the migration of the alarmAgents, following the pheromone concentration to reach UAVs.

Figure 5.3 illustrates an example of how an alarmAgent issued by a sensor node (Figure 5.3a) is routed through the network, following the pheromone trail (Figures 5.3a to 5.3d), until it reaches and delivers the alarm to a UAV (Figure 5.3e). This process is called trail-follow. The pheromone marks in the nodes are illustrated by numbers placed in the centre of the circles representing ground sensor nodes. The bigger the number, the stronger the pheromone mark. In this example, the number 10 represents the highest pheromone level, which represents a situation in which a sensor node just received a beacon from a UAV that is flying over it, while the number 0 represents the opposite situation, in which the sensor node has no pheromone mark. Notice that during the trail-follow, the alarmAgent can be redundantly sent by more than one node, according to the pheromone concentrations. This behaviour is achieved by using the same migrate-clone process used by the missionAgent while performing the mission dissemination, as presented in Section 3.2.

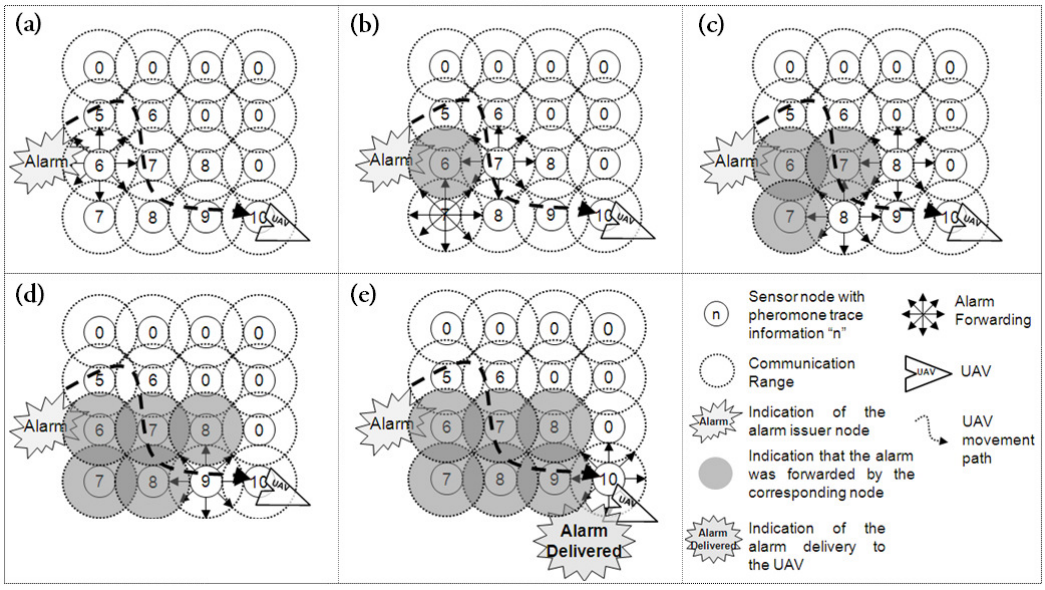


Figure 5.3: Pheromone-based alarm delivery example.

Aiming at robustness of the proposal, in case an alarm is issued by a node that has no pheromone trace ("0" pheromone mark on it), a direction is randomly chosen and the alarmAgent follows this direction until it finds a pheromone trail. When a pheromone mark is found in a node, it follows the respective trail as explained above. This situation is more likely to occur during system initialization, and in cases in which the number of

UAVs deployed in the system is very low and/or the size of the trail is small in relation to the size of the mission area. This mechanism is called trail-search, which is illustrated in Figure 5.4.

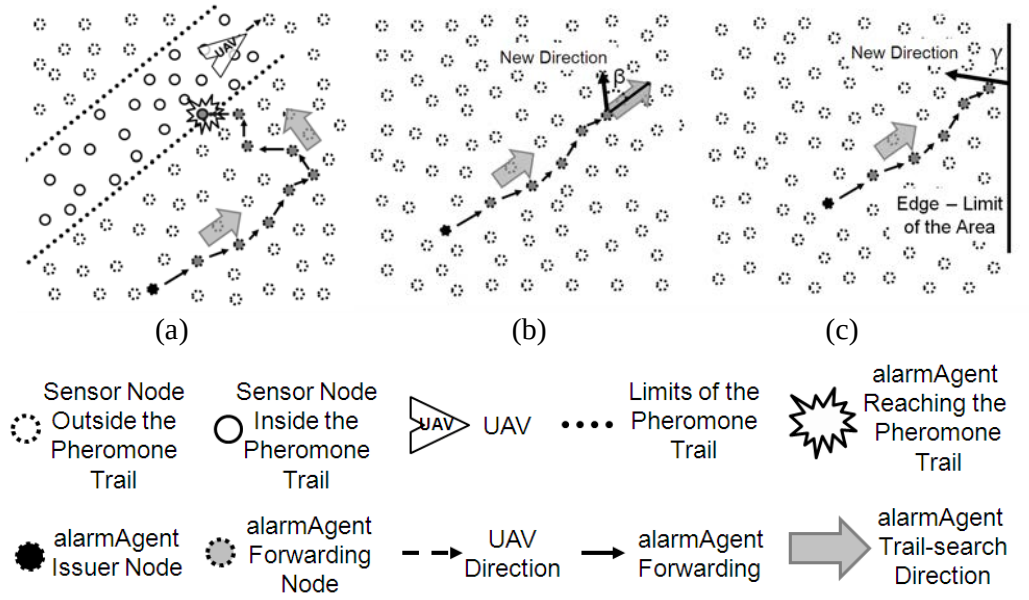


Figure 5.4: Trail-search mechanism: a) Illustration of the mechanism concept; b) General case for forwarding direction change; c) Particular case for forwarding direction change in the limits of the area.

As can be observed in Figure 5.4a, while performing the trail search, an alarmAgent follows the nodes towards a given direction, which is randomly chosen from the position of the alarm issuer node, until it reaches a trail or satisfies a given condition to change the direction of its forwarding. This condition can be defined as a number of hops or just by reaching the limits of the mission area, for instance. This movement of the alarmAgent is somewhat similar to the migration of the missionAgent in the dissemination phase presented in Section 3.2, but instead of having a defined MA as destination, the alarmAgent just follow a random direction. It is more similar to the migration performed by the beeAgent presented in Section 3.3.2 due to the randomness of the direction followed by the agent and to the avoidance of possible redundant duplication during the forwarding process.

Considering the direction which the alarmAgent follows in the trail-search, in the case in which the decision to change the forwarding direction is taken after a given number of hops, when arriving at a node that fulfils this condition, a new direction is chosen by the alarmAgent. This new direction is defined by an angle β randomly chosen according to a uniform distribution in the interval $(-\pi/2, \pi/2)$ in relation to the current direction. This is depicted in Figure 5.4b. If approaching the limits of the MA, by reaching a node that is located close to an edge that limits the area, a new direction in relation to this edge is chosen. This direction is defined by an angle γ randomly chosen according to a uniform distribution in the interval $(0, \pi)$, as shown in Figure 5.4c. Notice

that in the first case, in which the condition to change direction is determined by a number of hops, if the alarmAgent does not find a trail and reaches one of the edges that limits the MA, the same behaviour presented in Figure 5.4c is taken.

The implementation of the trail-search mechanism can be done by using a simple greedy position-based routing mechanism (STOJMENOVIC, 2002), as the one used by the missionAgent during the mission dissemination phase as presented in Section 3.2. The one adopted here considers the selection of the forwarding node based on the angle that the line that links the current node to a neighbour node forms with the reference direction. The neighbour node with which the current node forms the line that has the angle closer to the forwarding direction is selected to proceed with the alarm forwarding process. When an alarmAgent reaches a node that is placed in a position closer than one communication range from one of the edges that limit the MA, it is considered that the alarmAgent reached a limit edge, and thus a new angle is chosen as illustrated in Figure 5.4c.

The trail-search mechanism is also performed in cases in which the network is disconnected and while performing a trail-follow, the alarmAgent reaches a limit of a partition of the network and is not possible to proceed following the trail in the right direction. Then it starts a trail-search in an attempt to find another trail. While the alarmAgent is performing the trail-search towards a given direction, if it also reaches a situation in which it gets stuck due to network disconnection, it selects a new direction to proceed with the trail-search, acting similarly to what it does when it reaches a limiting edge of the mission area. This behaviour avoids deadlock situations.

The pheromone marks stored by the ground sensor nodes have three components: temporal, spatial and type classification. The first defines the elapsed time since the UAV beacon was received by the ground sensor node, while the second defines the distance between the UAV and the sensor node, which can be achieved by different methods, such as the received signal strength indication (RSSI) of the incoming beacon or by the current UAV's GPS position sent in the payload of the beacon message, for instance. The third component classifies the pheromone by a "flavour" ($\boldsymbol{F_p}$), which defines the type of threats that the UAV is able to handle, corresponding to the type $\boldsymbol{j}$ of the sensor that equips the UAV.

The first two of these components are used to define the pheromone concentration ($\boldsymbol{C_p(t)}$), which decays with the elapsed time since a ground sensor node receives a beacon from a UAV, as defined as follows:

$$\boldsymbol{C_p}(\boldsymbol{t}) = \boldsymbol{C_p}(\boldsymbol{t}-1)\cdot\boldsymbol{r} \;\Big|\; \boldsymbol{r}\in(0,1)\,. \tag{5.3}$$

where $\boldsymbol{r}$ is the tunable pheromone decay rate. This decay rate may have a predefined fixed value if all UAVs are assumed to have the same and constant speed; otherwise they transmit this decay rate to the ground sensor nodes in the payload of their beacon messages.

Infinity loops, in which the alarms would follow trails indefinitely, are not expected to happen since the alarms' propagation is much faster than the movement of mobile nodes, even in cases in which the mobile nodes move in closed paths (routes). This

because once a mobile node reaches the beginning of the path where the trails started, it will update the pheromone level of the static sensor nodes in that location and ahead, allowing the correct delivery of the alarms following that trail.

5.3.1 Pheromone Distribution over the Ground Sensor Nodes

The example presented in Figure 5.3 shows a uniform distribution of the ground sensor nodes. This simplifies the alarm forwarding process, as the pheromone information is evenly distributed among them, assuming for instance that the received signal strength indication (RSSI) is used to define the pheromone level. However, considering a more general case, in which the sensor nodes are randomly placed on the ground, such an even distribution of the pheromone information would hardly be achieved, which may cause problems in the alarm forwarding. The main problem is the increased number of sensor nodes that would forward a given alarmAgent in the trail-follow process, thus unnecessarily increasing the number of sent messages, leading to a waste of energy resources. To tackle this problem, a special region is defined in the centre of the pheromone trail, which constrains the broadcast of messages transmitting alarmAgents towards the UAV. To understand the reason for this, Figure 5.5a shows how an alarmAgent propagates in the trail without the definition of such region.

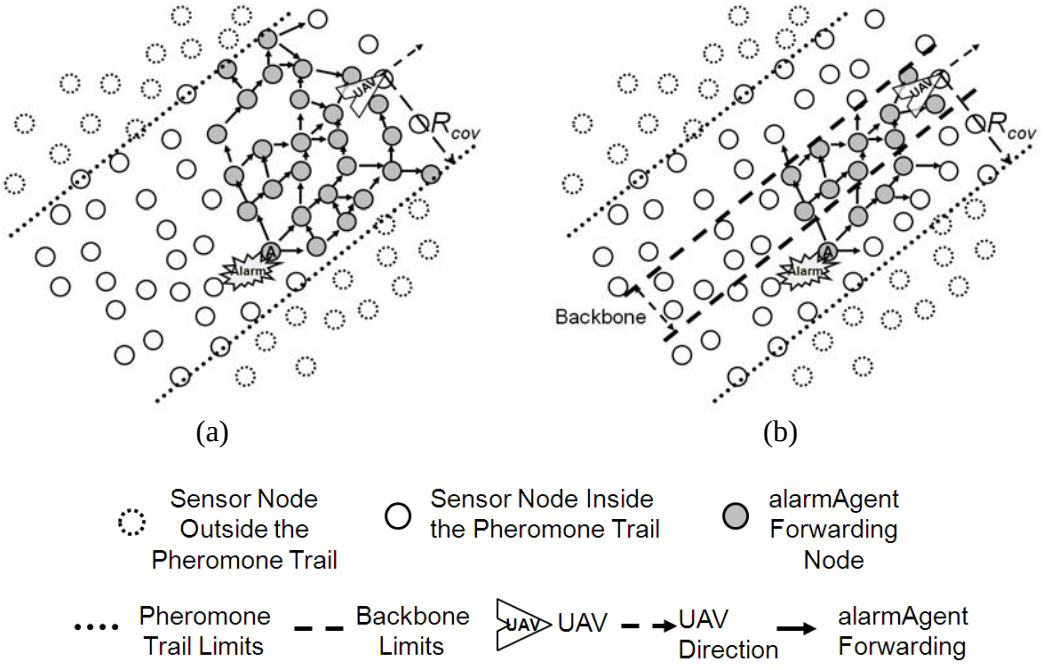


Figure 5.5: Alarm forwarding inside the pheromone trail: a) Trail-follow without backbone; b) Trail-follow with backbone.

As shown in Figure 5.5a, the alarmAgent is broadcasted towards the direction of the movement of the UAV, by creating clones that move towards the UAV by means of a migrate-clone mechanism as presented in Section 3.2. Notice that this generates a number of redundant forwarding alarm messages that are unnecessary. Using the spatial

component of the pheromone stored by the ground sensor nodes, it is possible to restrict the alarmAgent forwarding to the nodes closer to the real path followed by the UAV. Like this, the alarmAgent would be forwarded to the inner part of the trail, which is called the trail backbone, and, by reaching this inner part, the trail-follow can be constrained by its limits. The width of the backbone can be defined in terms of the UAVs' communication coverage range on the ground ($\boldsymbol{R_{cov}}$), and it can be wider or narrower according to the accepted level of redundancy. The reduction in the number of messages when using this backbone concept can be seen in Figure 5.5b.

Figure 5.6 visualizes the proposed backbone from a top-down two-dimensional perspective (Figure 5.6a) similar to Figure 5.5 and from a three-dimensional perspective (Figure 5.6b). In the figure it is possible to observe the UAV's communication range ($\boldsymbol{R_{com}}$), its coverage on the ground ($\boldsymbol{R_{cov}}$), and the delimitation of the backbone coverage ($\boldsymbol{R_{bb}}$).

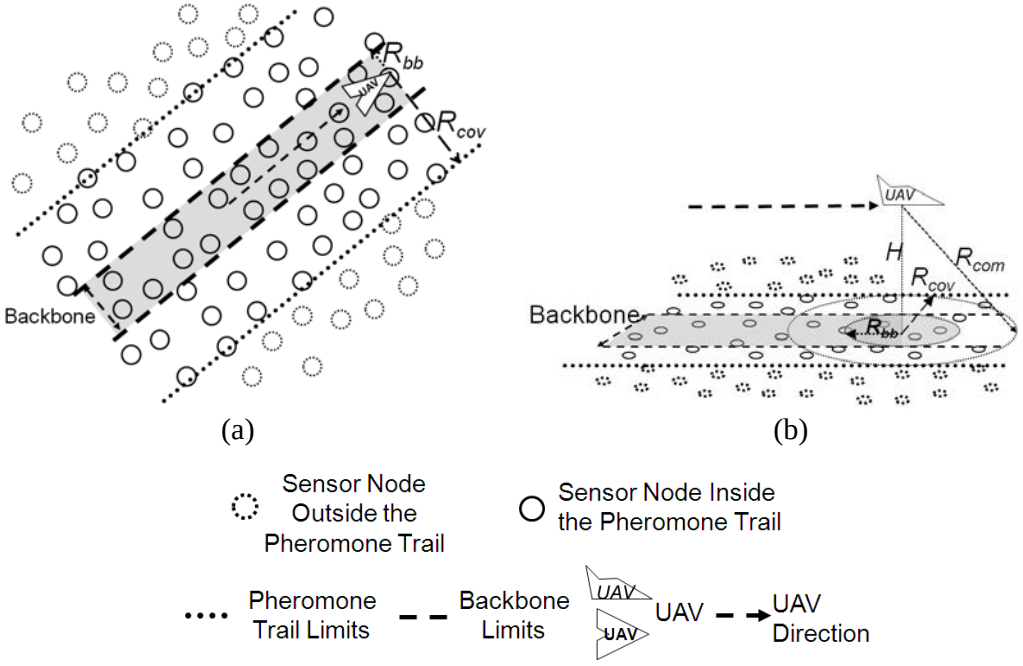


Figure 5.6: Pheromone trail with backbone: (a) top-down 2D view, (b) 3D view.

AlarmAgents issued by nodes located inside the backbone follow the backbone until they deliver their alarms to the corresponding UAV. AlarmAgents issued by nodes outside the backbone are first forwarded towards the backbone and then follow the backbone, as in the example of Figure 5.5b. This approach bounds the flooding of messages used to forward alarmAgents, i.e. the overhead of the trail-follow mechanism, to the limits of the backbone. This avoids the retransmission of alarmAgents by a large number of nodes, as it would be the case when a backbone is not adopted, as illustrated in Figure 5.5a.

5.3.2 Advanced Usage of Pheromones to Enhance Alarm Delivery

In the pheromone-based strategy presented above, when multiple UAVs fly over a given area, the sensor nodes located in that area take the pheromone information of each UAV and simply store them accordingly. In the occurrence of an alarmAgent that reaches these nodes while performing the trail-search, it will either follow the pheromone trail that points towards the direction of the closest UAV, i.e. the one that has the stronger pheromone concentration, or try to find the pheromone trail with the most suitable UAV to handle the corresponding event. This decision depends on how the pheromone concentrations are defined and on the type of pheromone, i.e. if the pheromone messages carry information about the sensors that equip the UAVs, the flavour mentioned before, or just indicate the UAVs' movement direction. If the UAVs' pheromones carry different flavours and they are analyzed to decide which trail an alarmAgent should follow, then heuristic-P is called ***heuristic-Pf***, in a reference to the flavour analysis.

Figure 5.7 illustrates the situation when a UAV crosses the trail of another UAV. In Figure 5.7a, UAV-1 leaves its pheromone marks creating its trail over sensor nodes on the ground, the same occurring for UAV-2. In Figure 5.7b, UAV-2 crosses the path followed by UAV-1, leaving its pheromone over some of the sensor nodes that form the trail of UAV-1. In Figure 5.7c, UAV-2 continues its path, and it is indicated that the nodes in the location where both trails cross have the pheromone information of both UAVs. Figure 5.7d presents almost the same information as Figure 5.7c, but it is possible to see more clearly that the nodes in each trail have the pheromone information of each UAV, and just those nodes in the intersection of the two trails have the pheromone marks of both UAVs.

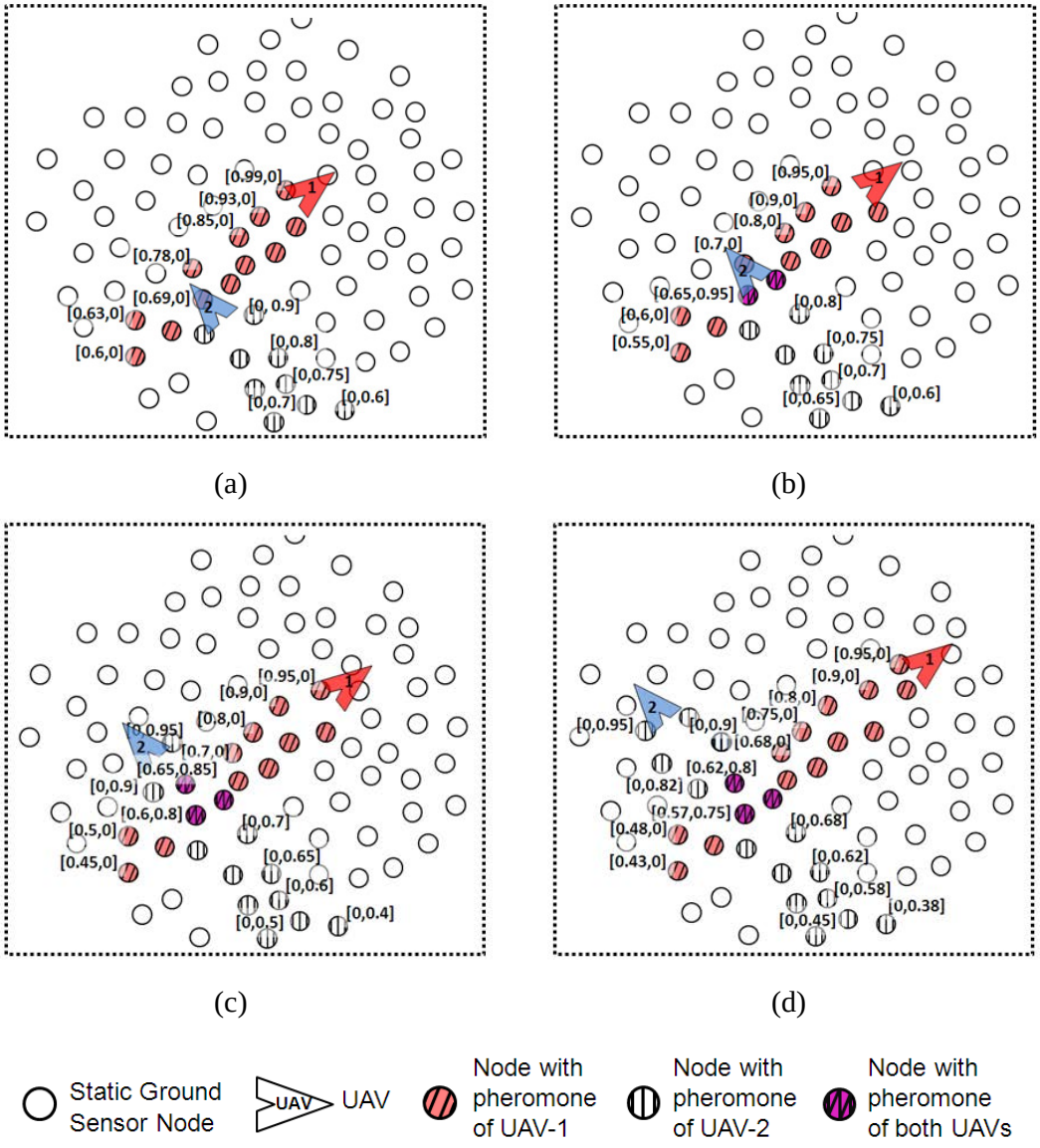


Figure 5.7: System behaviour when UAVs cross the path of one another.

It is also possible to observe in Figure 5.7, by following the numbers shown in squared brackets, the evolution of the pheromone concentrations on the ground sensor nodes related to each UAV. These numbers represent the pheromone marks for each UAV stored by the sensor nodes. In the figure, this information is shown only for some of the sensor nodes, as an example. The first element of the tuple represents the pheromone mark for UAV-1, while the second element corresponds to UAV-2.

A key feature for heuristic-P, and especially for the heuristic-Pf variation, is the way the pheromone information spreads itself through the ground sensor nodes. In principle, the sensor nodes get this information only by receiving the beacon messages from the respective UAV.

However, noticing that the overall system performance is heavily dependent on the pheromone information spreading and observing the situation highlighted in Figure 5.7, which describes the event of one UAV crossing the path of another, an opportunity to enhance the pheromone spreading mechanism was considered. At this point, it is important to understand why the spreading of pheromone information is so important to system performance. This is due to the fact that, as the system may be composed by UAVs carrying different types of sensors, some of them may be more suitable to handle a given event, while others may be less suitable or even incapable to do so. Then it is possible that an alarmAgent does not find a trail of a suitable UAV to handle it, or even no trail at all, and has to perform the trail-search until it finds such a trail. This trail-search mechanism may unnecessarily consume additional resources and, therefore, should be used only when strictly necessary. On the other hand, if the UAVs could collaborate to spread pheromones with different flavours, more trails of different flavours would be available, thus reducing the need for the random search based trail-search mechanism.

Observing the above mentioned facts in the situation presented in Figure 5.7, which is very likely to occur during system runtime, an approach to explore such a situation is proposed to enhance heuristic-Pf and to increase the overall system efficiency. This proposal aims to improve the dissemination of the pheromones' spread by the UAVs and considers three consecutive improvements: 1) Pheromone hitchhiking (***Heuristic-Pf-h***); 2) Pheromone hitchhiker backwards dissemination (***Heuristic-Pf-hb***); and 3) Pheromone dissemination in both trails (***Heuristic-Pf-hbt***).

5.3.2.1 Pheromone hitchhiking (Heuristic-Pf-h)

As a UAV crosses the path of another one, for instance UAV-2 crosses the path of UAV-1 in Figure 5.7, it can be informed by the ground sensor nodes about the previous UAV that passed over the area. Like this, it may take the pheromone information of this previous UAV and spread this information together with its own pheromone. It is possible to state that the pheromone of one UAV is getting a "ride" on the other UAV (thus the use of the term "hitchhiking" to define this enhancement). Acting this way, every node that receives the pheromone of the second UAV will also receive the information of the first one. This enhancement in the pheromone dissemination is especially important when UAVs have different capabilities, which make them able to handle different types of events. Figure 5.8 presents the same situation described in Figure 5.7, but using this enhancement.

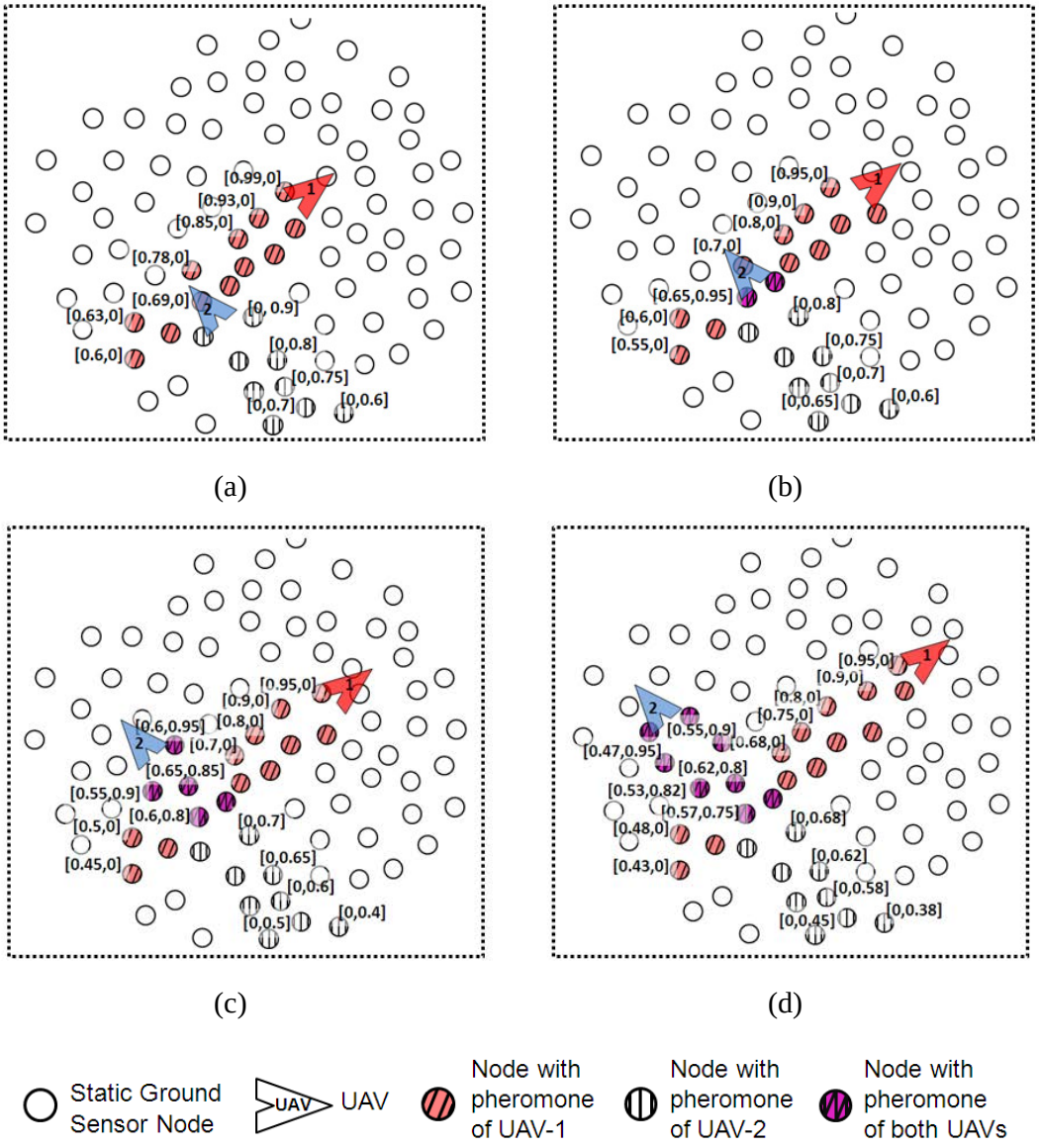


Figure 5.8: Pheromone hitchhiking.

Figures 5.8a and 5.8b present the same situations as in Figure 5.7. The difference can be observed in parts 5.8c and 5.8d, which show UAV-2 spreading both pheromone information (its own and the "hitchhiker" one from UAV-1) to the sensor nodes in its path after crossing the nodes that form the intersection with the pheromone trail of UAV-1. This mechanism is efficient, as it does not incur any additional overhead in the system. As UAV-2 has to send beacon messages with its own information, the spread of the pheromone information of UAV-1 "takes a ride" in these beacons, so it is disseminated "for free". Only one additional message is required: the message that UAV-2 receives from sensors in the UAV-1 trail when it flies over them, in order to get knowledge about the pheromone of UAV-1 in this area and its current level. Notice that this is an important detail about the mechanism. As there is an elapsed time from the

moment when UAV-1 has passed over the area which UAV-2 is currently crossing, the pheromone of UAV-1 has already started to "evaporate" according to the decay in (5.3). This means that the information that UAV-2 will spread about UAV-1 should correspond to such a situation, so that the correct concentration of this pheromone is informed. This can be observed in Figures 5.8c and 5.8d, in which the nodes in the UAV-2's path are receiving decreasing amounts of pheromone of UAV-1 from the beacons of UAV-2. The numbers in the tuples present this information, for instance, the tuples for the two sensor nodes that are closer to the backend of UAV-2, which have the tuples [0.47, 0.95] (the one on the left) and [0.55, 0.9] (the one on the right). Moreover, just the sensors in the backbone of the trail send hitchhiker pheromone information to the UAVs, as they are the ones that have the stronger pheromone level in a trail.

5.3.2.2 Pheromone hitchhiker backwards dissemination (Heuristic-Pf-hb)

As already mentioned, the spread of pheromone information is essential for the efficiency of the designed alarm delivery approach. The first proposed enhancement described above does really help in this task, as the UAVs help each other to spread their pheromone information. However, as presented in Figure 5.8, only the nodes ahead in the UAV-2 path will receive the information about UAV-1. It would be good if all nodes in the UAV-2 trail could have such information.

A possible solution for this problem can be obtained by making the sensor nodes spread the information of the first UAV (or previous UAVs) that passed in the area when they notice that they are making part of a new trail, i.e. a trail of another UAV. This mechanism will push the pheromone of the previous UAV(s) backwards towards the trail of the UAV that is currently flying over the area. Figure 5.9 presents this situation. In this figure, parts (a), (b) and (c) show the communication that disseminates the information backwards in the trail of UAV-2, while Figure 5.9d presents the final situation that will eventually emerge, when all nodes in the trail of UAV-2 will have also the information about UAV-1. In order to provide a clear view of the situation, the tuples with the pheromone information were suppressed in Figure 5.9.

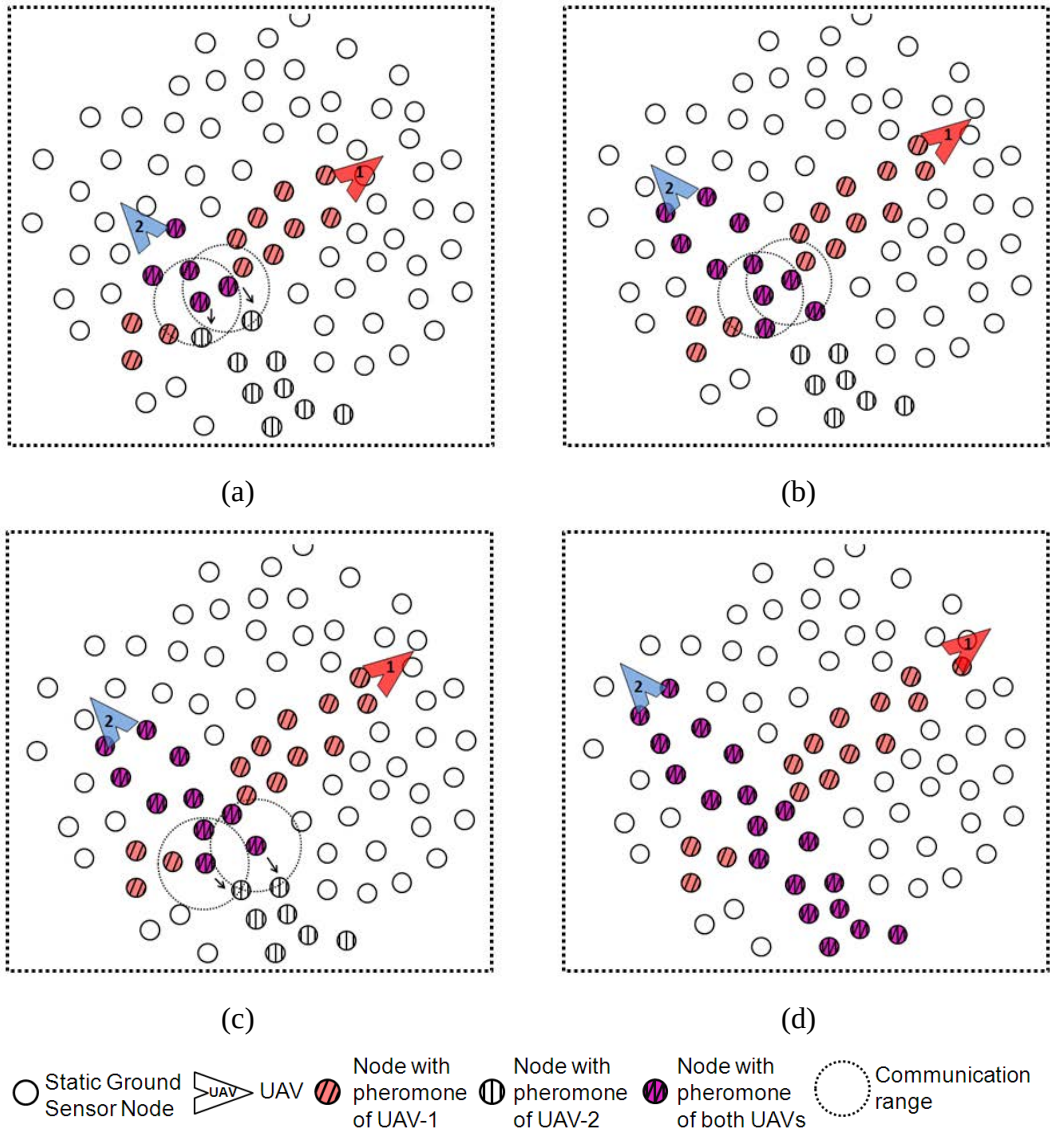


Figure 5.9: Pheromone backwards dissemination in the UAV trail.

5.3.2.3 Pheromone dissemination in both trails (Heuristic-Pf-hbt)

Following the reasoning that led to the idea of spreading the pheromone backwards into the UAVs' trails, a natural extension of this proposal would be also to spread the pheromone information of a given UAV into the trail of the UAV that has previously passed over the area. The mechanism used to perform this task would be similar to the one presented before. When sensor nodes notice that another UAV is crossing the area where there is a trail of a previous one, they forward the new pheromone information to the nodes belonging to the first trail. This situation is presented in Figure 5.10, in which it is possible to observe in parts (a), (b) and (c) the dissemination of the information to nodes that belong to the trails of both UAVs. Figure 5.10d presents the situation that

eventually emerges, when all nodes in both trails have the information about both UAVs.

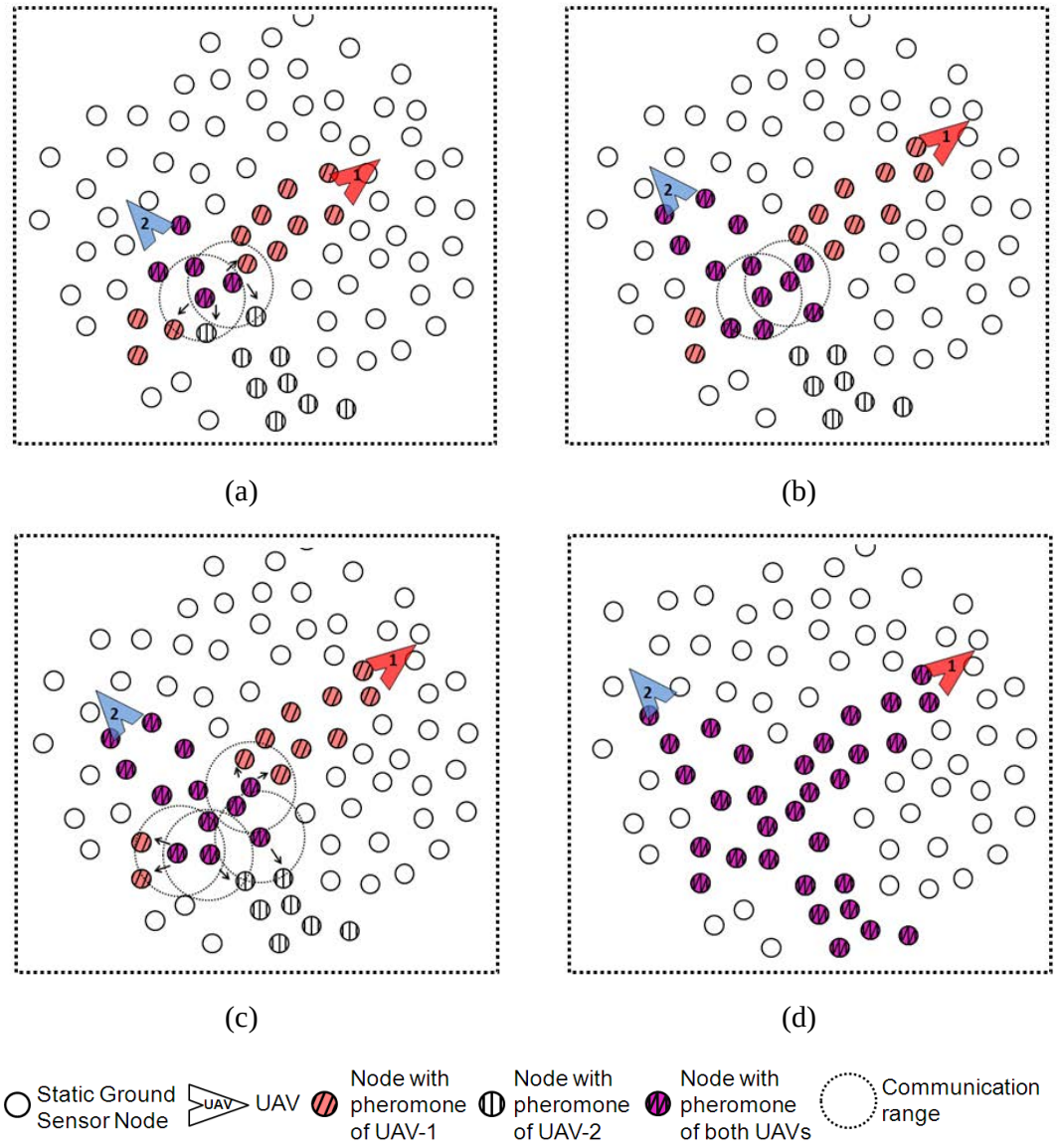


Figure 5.10: Pheromone dissemination to two UAV trails.

Additionally, as it can be observed in Figure 5.10d, the pheromone information of UAV-2 comes to a node nearby UAV-1. So, a further enhancement proposed is that this UAV takes the information of UAV-2 and, in a way similar to UAV-2 helping to spread its pheromone information, UAV-1 will get the pheromone information about UAV-2 and also helps in its dissemination. This approach is depicted in Figure 5.11. Figure 5.11a highlights the arrival of the information about UAV-2 to UAV-1 via the ground sensor nodes (an arrow is included in the figure to point out the nodes where this aspect

can be observed). Figure 5.11b shows UAV-1 leaving the pheromone of both UAVs instead of only its own. An arrow highlights this event.

It is important to observe that these last two proposed extensions, which use the dissemination of the pheromones via the ground sensor nodes, may imply in an additional overhead. Because of this, they can be seen as optional features that are applied depending on the available resource budget. There is a trade-off between the efficiency gain and the use of required resources. This trade-off will be further explored in the analysis of the experimental results in Section 5.5.

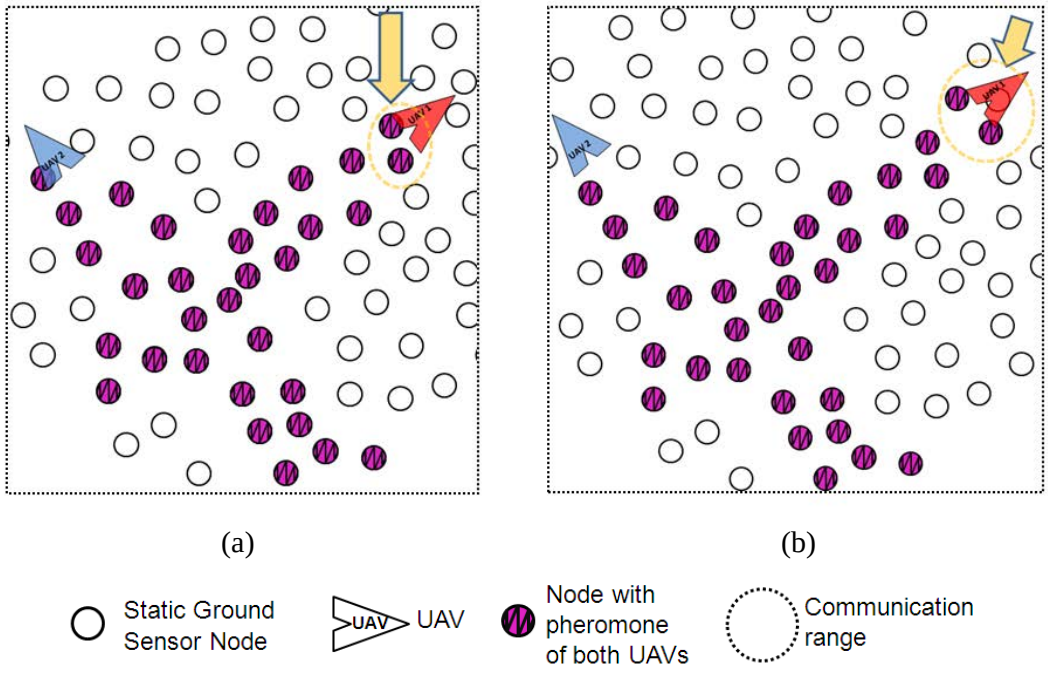


Figure 5.11: Pheromone dissemination of both UAVs by each other.

5.4 Feasibility Analysis

This section presents a feasibility analysis of the proposed pheromone-based mechanism for the cooperation among mobile sensors in the air and static sensors on the ground. It is assumed that a mobile sensor node, a UAV u_i, flies at a certain constant altitude H and with a constant speed v_{ui}. It has a communication range R_{com}, which provides communication coverage for a circular area with radius R_{cov} on the ground, as presented in Figure 5.12 and determined by:

$$R_{\text{cov}} = \sqrt{R_{com}^2 - H^2} \quad . \tag{5.4}$$

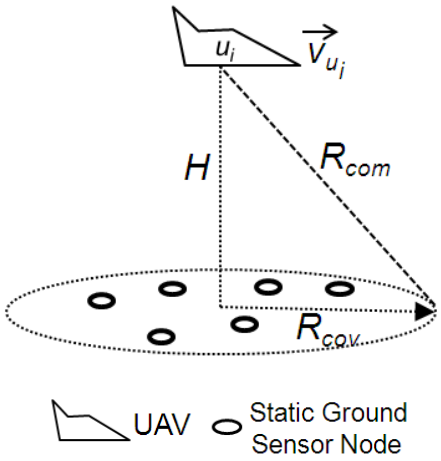


Figure 5.12: Area with radius R_{cov} covered on the ground by help of radio communication with range R_{com}.

It is also assumed that the UAVs move according to a random mobility pattern based on the study presented in (SPYROPOULOS; PSOUNIS; RAGHAVENDRA, 2006), where the trajectory followed by a UAV is composed of a sequence of steps. At the beginning of each step, a UAV randomly chooses a direction defined by an angle ψ (its heading angle explained in Section 5.2) in (0, 2π) and constrained to a subinterval between these values according to the particular characteristics of the different UAVs. Based on its current coordinates (x_{ui},y_{ui}) and the chosen angle ψ, it determines the goal coordinates (x'_{ui},y'_{ui}) for this step. Then a vector from the current coordinates to the goal ones is drawn, which has a length L_j, which is an exponentially distributed random variable. Distributions of the different lengths L_j (j = 0, … N | N is an integer) are independent and identically distributed (IID) random variables, having a common average. Figure 5.13 presents the elements described above, in which it is possible to observe the coordinates of the current UAV at point A, the goal coordinates at point B, and the corresponding vector with length L_j linking these two points.

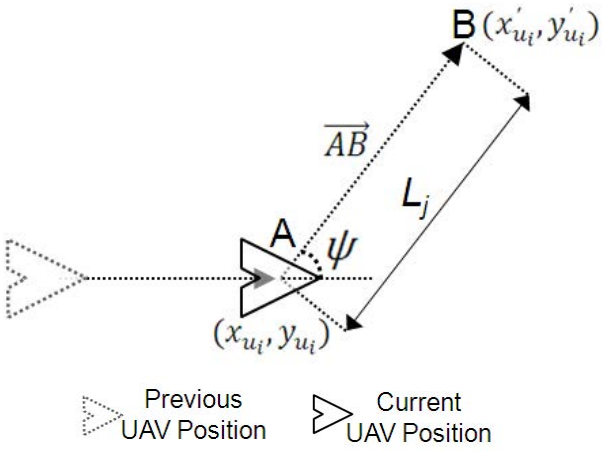


Figure 5.13: Elements to determine the random movement followed by the UAVs.

Considering the model of the random movement followed by the UAV and its communication coverage on the ground, it is possible to determine the timing requirements for the ground sensor nodes when receiving the beacons sent by a UAV. These timing requirements are used to determine the minimum period that should be used by the UAVs to send sequential beacons. Figure 5.14 presents the general case for the corresponding movement and the involved variables. Assuming that a UAV takes Δt_j time units to move from a given position A to another position B (with a length L_j between A and B), and that it moves with constant speed, then Δt_j is calculated as:

$$\Delta t_j = \frac{L_j}{\left|\vec{v}_{u_i}\right|} \ . \tag{5.5}$$

It is assumed that a beacon is composed by a single communication packet, for which a delay between transmission and reception is negligible. The time Δt_j is the interval in which the UAV has to send such a beacon to a ground sensor node within a step of its movement. Moreover, depending on the variability factors selected for the random choices of the goal coordinates and of the angle that defines the random movement, L_j can be bigger or smaller than the diameter of the area covered by the UAV communication. This has impact on Δt_j, and considering the average for the values that L_j may assume, an average for Δt_j can be determined as follows:

$$\overline{\Delta t} = \begin{cases} \dfrac{\overline{L}}{\left|\vec{v}_{u_i}\right|}, & \textit{if} \quad 0 \le \overline{L} \le 2R_{Cov} \\ \dfrac{2R_{Cov}}{\left|\vec{v}_{u_i}\right|}, & \textit{if} \quad \overline{L} > 2R_{Cov} \end{cases} . \tag{5.6}$$

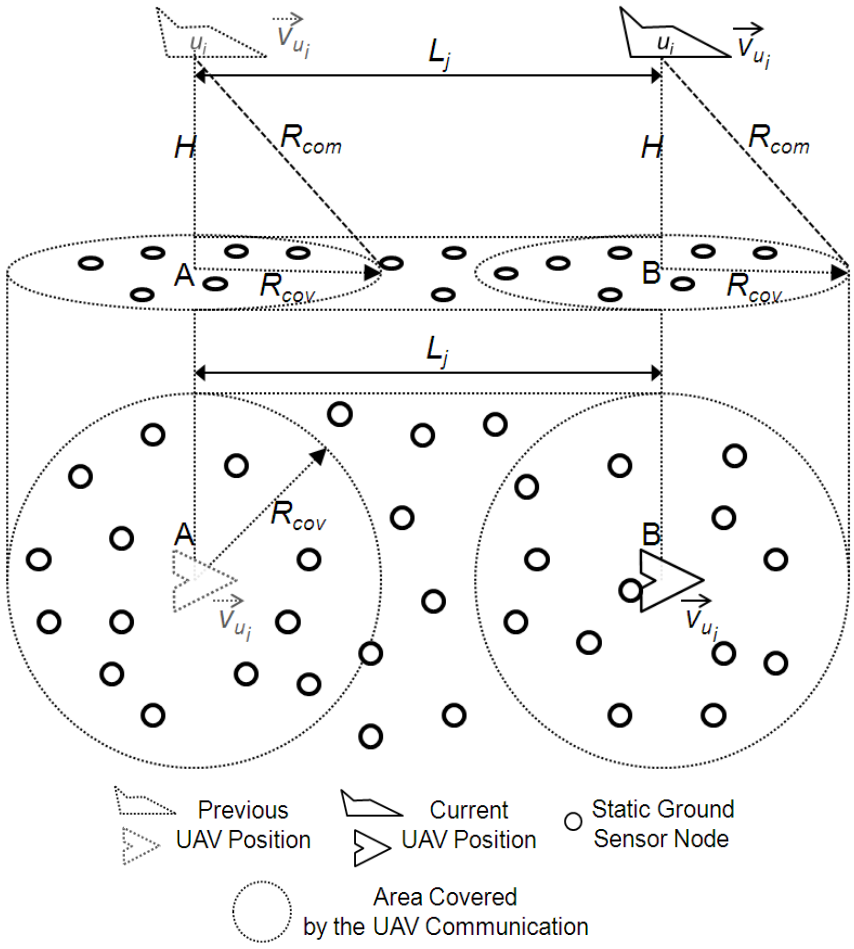


Figure 5.14: General case of the UAV movement and relationship between the communication range and the movement step.

Considering small UAVs having communication capabilities provided by COTS devices such as XBee-Pro (DIGI, 2009) that implements the IEEE 802.15.4 standard with extended range (IEEE, 2003), and assuming reasonable values for $\boldsymbol{R_{com}}$ = 1Km, $\boldsymbol{H}$ = 250m, and $\boldsymbol{v_{ui}}$ = 100 Km/h (UVS, 2009), a value around at least 36 seconds results for the periods according to (5.6), which would be an upper bound value for these periods.

If the current GPS position is sent within the beacon message, the UAVs can send one beacon per period (period equal to the time interval calculated by (5.6)) that it is guaranteed that all the nodes in their path will receive the beacon and will have their correct pheromone information. However, assuming that the sensor nodes on the ground do not receive the GPS position of the UAV in the beacon, they calculate the pheromone level based on the signal strength of the received beacon message (RSSI). Thus, if a UAV sends just one beacon at each movement step, then, using a period between beacons resulting from (5.6), it is possible that the alarmAgent gets mislead. This may occur because the pheromone information on the sensor nodes on the ground may point to the inverse direction of the UAV movement. This is possible if a beacon is

sent, for instance, in the beginning of the UAV's movement step, so that a node immediately below the UAV will have a stronger pheromone mark if compared to the mark stored on those nodes located ahead of the UAV's current position (especially those in the border line of the UAV's communication range). In a situation like this, if these sensor nodes do not receive any other beacon updating their pheromone information, they will indicate the wrong direction of the UAV. Figure 5.15 presents this situation, in which it is possible to observe in Figure 5.15a the UAV in the beginning of the movement step, sending a beacon which is stored as pheromone mark accordingly by the sensor nodes on the ground. However, observing Figure 5.15b, it is possible to notice that, if the sensor nodes do not receive a new beacon updating their pheromone information before the end of the UAV movement step, they will indicate an erroneous direction of the UAV. Besides, it can even occur situations in which a number of sensor nodes stay without any pheromone mark at all, as shown in the figure.

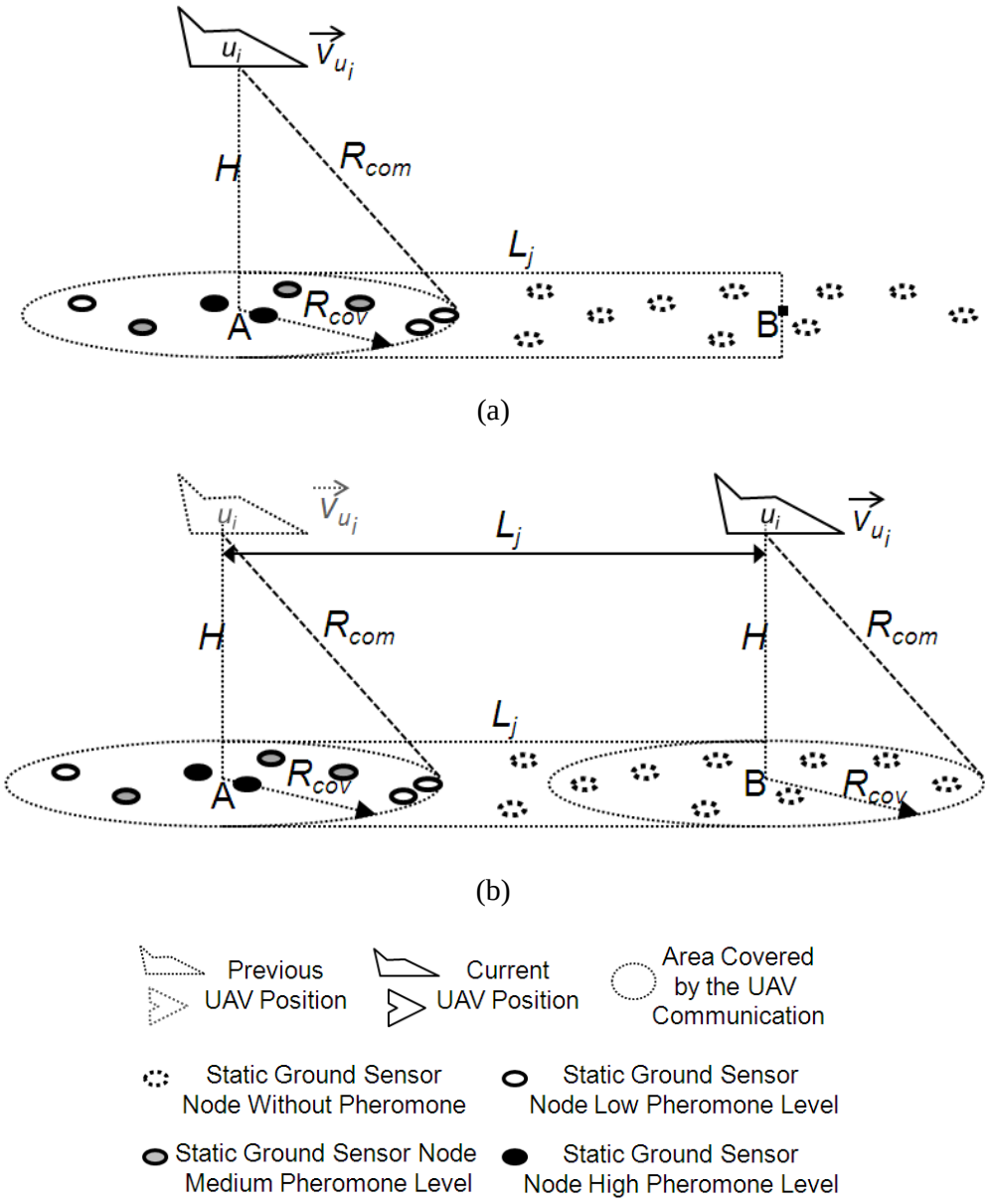


Figure 5.15: Misleading pheromone information. (a) Beginning of the movement step; (b) Final of the movement step.

To handle the above mentioned situation, it is necessary to assure that the UAVs send beacons in such a way that the pheromone information stored by the ground sensor nodes is correctly updated according to their movement. This can be done by adjusting the period in which the beacons are sent (t_b), which will not be equal to the result obtained in (5.6). Observing the conditions presented in (5.6), and the limit situation in which $L_j = 2R_{cov}$, if a UAV sends a beacon after flying a distance corresponding to R_{cov} from the beginning of its movement step (position A in Figures 5.14 and 5.15), the ground sensor nodes behind its current position will have their pheromone information

correctly updated. Figure 5.16 leaves this situation clear, in which it is possible to observe that the UAV should then send a new beacon message at least at the positions B, C, D and E, after the first beacon sent at position A, thus after flying a distance corresponding to $\boldsymbol{R_{cov}}$.

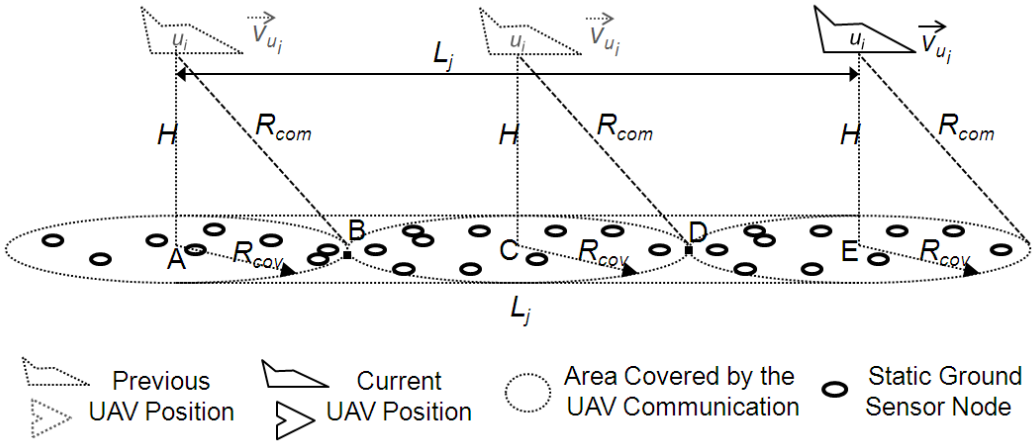


Figure 5.16: Limit situation in which $\boldsymbol{L_j = 2R_{cov}}$.

By the analysis presented so far and considering the conditions presented in (5.6), the period between beacons $\boldsymbol{t_b}$ can be calculated by:

$$t_b = \begin{cases} \dfrac{\overline{\Delta t}}{2}, & \textit{if} \quad 0 \le \bar{L} \le 2R_{Cov} \\ \dfrac{\overline{\Delta t}}{\left\lceil \bar{L} / R_{Cov} \right\rceil}, & \textit{if} \quad \bar{L} > 2R_{Cov} \end{cases} . \qquad (7)$$

Taking the same parameters used for the calculation of a possible value of the period between beacons (around 36 seconds) previously presented, $\boldsymbol{t_b}$ would have a value around 18 seconds. It is thus possible to state that the required timing conditions for beacon emissions are very relaxed and easy to be fulfilled. This indicates the feasibility of the proposed mechanism.

This above situation takes into account that the pheromone level stored by the ground sensor nodes is calculated exclusively by, for instance, the RSSI of the beacon message sent by the UAV and by the time elapsed since the sensor nodes received the beacon. However, considering further the usage of the pheromone backbone, as explained in Section 5.3.1, another constraint has to be taken into account, which is the radius that defines the backbone ($\boldsymbol{R_{bb}}$). Considering that $\boldsymbol{R_{bb}}$ is a fraction of $\boldsymbol{R_{cov}}$, Figure 5.17 presents the same situation described in Figure 5.16, but with the addition of the

radius of the backbone, which is represented by the shaded circle with radius R_{bb}, concentric with the one formed by R_{cov}.

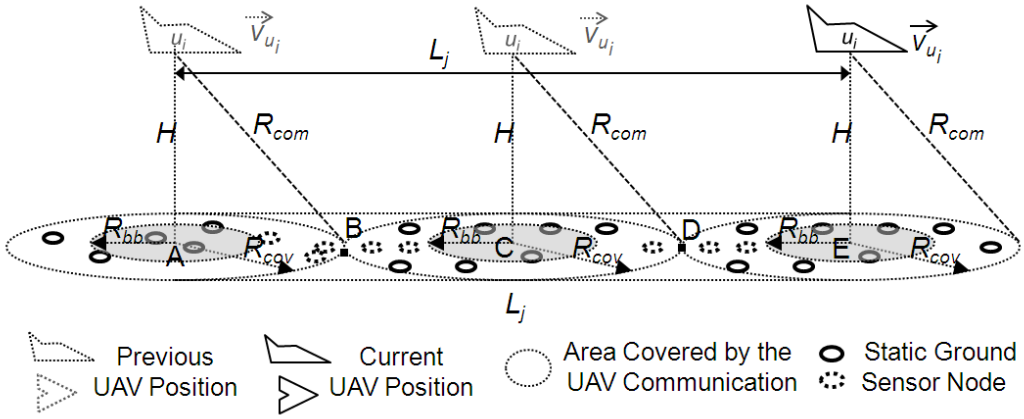


Figure 5.17: Communication coverage for the limit case considering the backbone layer.

Considering that all sensor nodes inside the backbone have pheromone levels higher than those of the nodes outside the backbone, a similar analysis shows that there are nodes that should be covered by the shaded area to form the backbone, but they are not. This is the case of the nodes represented with dotted lines in Figure 5.17, which could occur even if the UAVs sent beacons at positions B and D. Depending on the rate R_{bb}/R_{Cov} the condition sufficient to validate (5.7) is not enough to assure that the correct levels of pheromone will be stored by the ground sensor nodes in the backbone. If $R_{bb}/R_{Cov} \geq 1/2$, (5.7) is still valid as the sensors on the path of the UAV will be correctly covered. However, an additional constraint for the case in which $R_{bb}/R_{Cov} < 1/2$ has to be added to calculate t_b:

$$t_b = \begin{cases} \dfrac{\overline{\Delta t}}{\left\lceil \overline{L} / R_{Cov} \right\rceil}, & \textit{if} \quad R_{bb}/R_{Cov} < 1/2 \\ (7), & \textit{otherwise} \end{cases} . \qquad (8)$$

Assuming values of R_{bb} as small as 1/10 of R_{cov}, and using the same parameters presented in the calculation of t_b above, the new values for t_b would be around 2 seconds, which is still a reasonable condition to be fulfilled.

From the perspective of the ground sensor nodes, it should be considered that they alternate between two states, the active and sleep states, which represent the duty cycling mechanism implemented to preserve energy resources as mentioned in their model presented in Section 5.2. The nodes sleep most of the time and wake up at regular intervals (t_w) and then stay awake for a short time to listen for possible communications, transmit some data, and/or perform data sampling. In case in which

there is no communication activity or the sampled data does not require further processing, they come back to the sleep mode for another t_w interval.

Observing that $t_b >> t_w$ (t_b is orders of magnitude larger than t_w) (YUE; SUN; JIN, 2010) (KUMAR et al., 2010), besides the above analysis it is also important to assure that the sensor nodes are awake when the beacon messages are sent by the UAVs flying over them, so that they can be received. This is possible by making the preamble of the beacon message be at least as long as the time interval in which the sensor nodes wake up, i.e. t_w, which is a solved problem by using for example the B-MAC protocol as MAC layer (POLASTRE; HILL; CULLER, 2004).

5.5 Experiments and Results

Results from simulations of the described bio-inspired approach are presented for different setups so that both system efficiency (in terms quality factors of the alarm handling) and overhead could be evaluated. Basic simulation parameters are first presented and hold for all simulated setups, while the variations used to access specific aspects are presented along the subsections according to the specific setups.

5.5.1 Basic Simulation Parameters

Based on the scenario presented in Section 1.2.3, the simulated environment was defined as a squared area in which static ground sensor nodes are randomly deployed with independent uniform probability (homogeneous Poisson point process in two dimensions, which generates a geometrical random graph). These sensor nodes have a communication range of 350 meters and were deployed with a density of 50 nodes/Km^2 for all tested sizes of areas, providing a connected network. This is an important assumption, as gaps in the network connectivity negatively impact the results, since these gaps may stop the alarm forwarding process. The problems related to these gaps are further discussed in Chapter 8.

The trail-search mechanism is set to choose a random direction ($0, 2\pi$) according to a horizontal reference from the alarm issuer nodes, and to change its direction when an alarmAgent reaches the edges of the mission area, choosing an angle in the interval ($0, \pi$) in relation to the edge.

UAVs patrol the area flying at an altitude of 250 meters according to the movement pattern described in Section 5.4, with an average movement step length equal to 500 meters and heading angle ψ in ($0, 2\pi$) and at 100 Km/h speed. Their communication range (R_{com}) is 500 meters. The choice of these setup parameters are based on the characteristics of the scenario analyzed in this work and the mobility model described in Section 5.4, considering the usage of small (Mini or Micro) UAVs that have operational range of 10Km (UVS, 2009), using COTS communication technologies such as those based on IEEE 802.15.4 (extended range version) radios. Unless stated differently, the simulations of the pheromone-based strategy use a backbone range (R_{bb}) equal to half of the UAVs' communication coverage range on the ground (R_{cov}). This condition allows the use of (5.7) to calculate the pheromone beacon sending period. The pheromone decay rate r, is established to 0.995 units per second, and, after 180 seconds, the sensor

nodes consider that the pheromone mark disappears, i.e. its level is set to 0. The described parameters provide trails with width close to 1Km and 5 Km of maximum length, which is reached when the UAV flies in straight line for a period equal to the trail lifetime (180 seconds).

The simulations were divided into different sets, according to variations that allow the assessment of different features and considering a corresponding goal for each set. Additionally, 100 runs were performed for each set.

Squared areas with five different dimensions were used as test scenarios for the simulations, namely 2Km × 2Km, 4Km × 4Km, 6 Km × 6Km, 8Km × 8Km and 10Km × 10Km. Figure 5.18 presents those areas in scale for comparison reasons.

Each area presented in Figure 5.18 illustrates a pheromone trail in scale with dimensions described by the above parameters used for the simulations. These illustrations are useful to provide a visual perspective of how much of each area can be covered at most by a trail of one UAV, in the case in which it is flying in a straight line. The gray gradient of the trails in Figure 5.18 represents the strength of the pheromone level stored by the ground sensor nodes in the corresponding positions, in which the lighter the gray is, the weaker is the pheromone level.

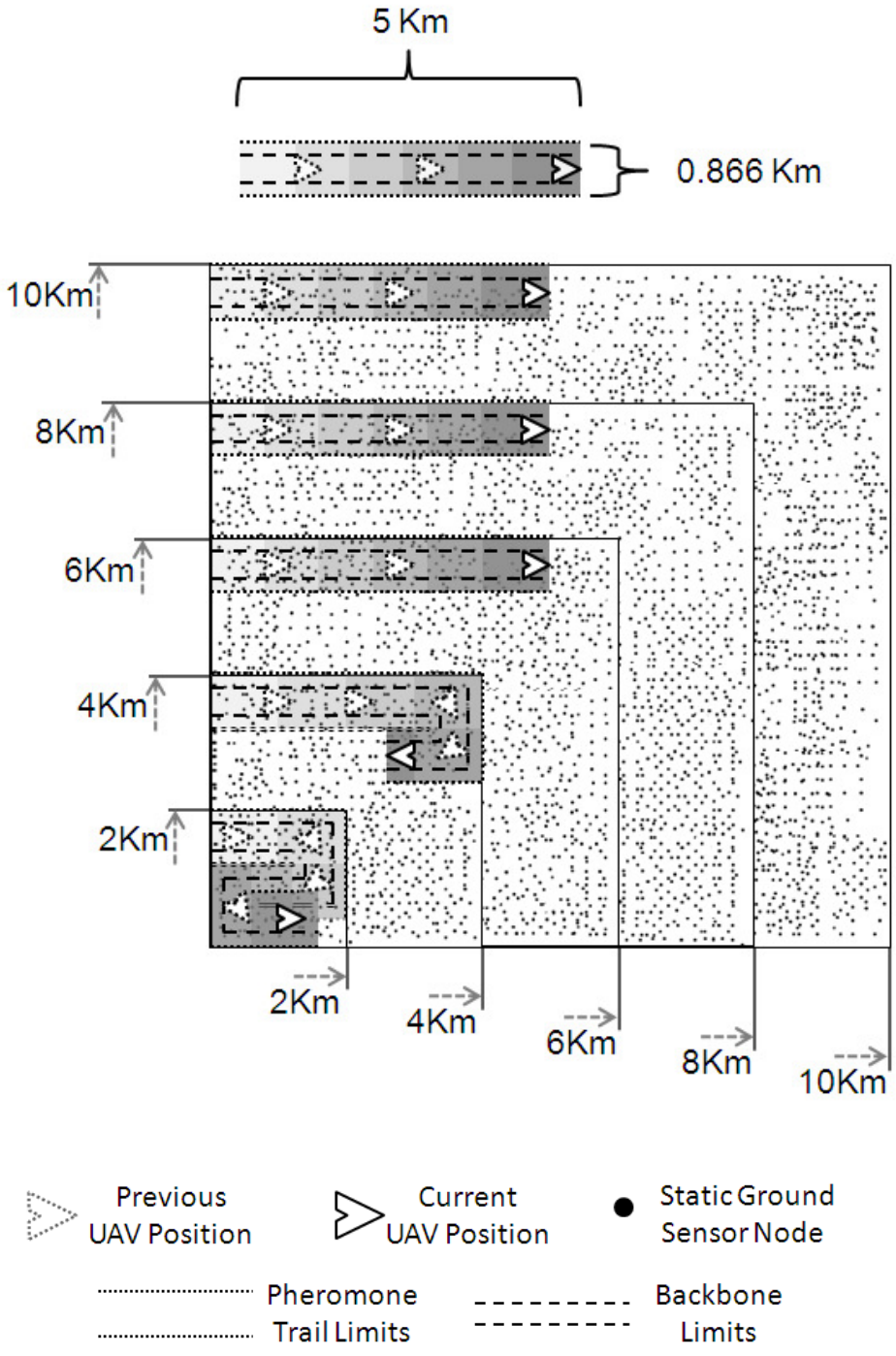


Figure 5.18: Examples of the areas used as test case scenarios for simulations and the bigger area that a trail can cover.

5.5.2 Assessment of the Basic Pheromone Mechanism

The first set of simulations has the goal of assessing the usefulness and efficiency of the basic pheromone strategy (heuristic-P) with the backbone in the trails to deliver alarms to the UAVs. These simulations consider all UAVs having the same capabilities to handle threats, i.e. they all have sensors of the same type. Different numbers of UAVs and threats are considered.

Two metrics are assessed; the first and most important one is related to the cost associated with the alarm delivery, which is calculated in terms of the total number of messages that were sent during each simulation run to deliver the alarms to the UAVs. The results presented are average values based on hundred simulation runs for each average. The second metric measures the elapsed time from the moment in which the alarm is issued until the instant in which the UAV that will be responsible for handling the alarm arrives at the location where the threat was detected (from now on called "time-to-handle" metric). This metric is an average of the time needed to handle alarms per threat from a set of 100 simulations runs.

The achieved results are compared with both an optimum and a flooding-based solution. The optimum (minimum hop) solution considers the minimum number of hops from the alarm issuer node to the closest UAV, thus minimizing the cost in terms of the number of messages that are transmitted in the network to deliver an alarm, and then it is called Optimum-C. This optimal solution is implemented by using the global state of the simulations, which gives access to the positions of all sensor nodes and UAVs, needed to choose the shortest path from the alarm issuer node to the closest UAV. In practice, the implementation of an equivalent solution would imply a very large overhead, as each time the movement of a UAV changes its connections to the ground sensor nodes, a flooding to the whole network would be needed to update the new routes in all the sensor nodes. The flooding-based solution considers that an alarm is forwarded by sensor nodes in all directions from the position of the alarm issuer node until the alarm reaches a UAV, instead of being carried by an alarmAgent that takes intelligent decisions to reach the UAV as presented by the proposed pheromone-based approach.

A squared area of 4Km × 4Km was considered as scenario for this first set of simulations. The scenario was simulated with the combinations of three numbers of UAVs (1, 2 and 4), and four different numbers of threats (1, 3, 5 and 7) appearing at the area according to a uniform random distribution within a time window of 180 seconds, which is the pheromone trail lifetime established by the parameters presented above. The rationale for this specific arrival model is to analyze the disturbances that this increasing number of threats may induce in the achieved results in relation to the pheromone trail lifetime. Consistently with this goal, these specific numbers of threats were chosen to stress these possible effects according to the tested numbers of UAVs.

In Figure 5.19 it is possible to observe the large difference in results provided by an ordinary flooding-based solution, the proposed pheromone-based one and the optimal reference solution. This big difference motivated the use of semi-log graphs in Figures 5.19a to 5.19c, so that this difference could be observed. Moreover, it is also possible to observe that the proposed solution presents a fairly good efficiency if compared to the

optimum one. Besides, the proposed solution scales with the increasing number of appearing threats, as it is shown in Figure 5.19d. Results in Figure 5.19d indicate that the increase in the number of employed UAVs reduces the amount of messages sent by a rate of 20% in average in relation to the closer setup with fewer UAVs, i.e. this reduction is seen when the setup with 2 UAVs is compared to the one with 1 UAV and the one with 4 UAVs is compared to the one with 2 UAVs. Table 5.1 complements Figure 5.19 providing the values for the averages plotted in the figure followed by plus-minus the value of the respective standard deviation.

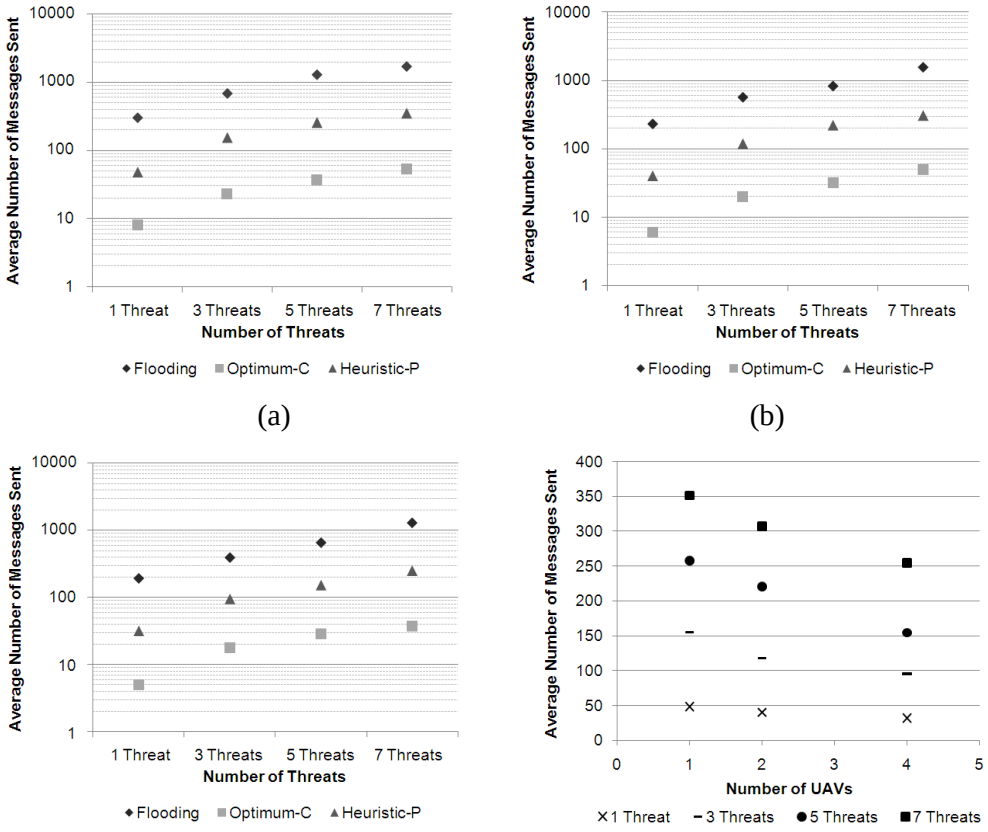


Figure 5.19: Cost assessment in terms of number of messages sent for the different setups with: a) 1 UAV; b) 2 UAVs; c) 4 UAVs; d) comparison among the different numbers of UAVs.

Table 5.1: Average and standard deviation values for the number of messages sent.

Number of Threats	Heuristic-P			Optimum-C			Flooding		
	1 UAV	2 UAVs	4 UAVs	1 UAV	2 UAVs	4 UAVs	1 UAV	2 UAVs	4 UAVs
1 Threat	48.6 ±11.03	40.12 ±12.23	31.72 ±9.67	8.21 ±2.13	6.73 ±1.97	5.51 ±1.45	305.66 ±139.43	230.57 ±107.54	192.31 ±75.93
3 Threats	155.03 ±24.01	118.01 ±20.02	95.54 ±15.66	23.12 ±5.18	20.48 ±3.83	18.51 ±2.97	691 ±277.12	566 ±213.45	391.87 ±122.36
5 Threats	257.5 ±35.65	220.29 ±30.01	154.57 ±26.05	37.45 ±7.24	32.26 ±5.73	29.72 ±5.09	1301 ±409.83	827 ±230.52	652.39 ±155.67
7 Threats	315.09 ±45.14	306.63 ±40.08	254.34 ±34.51	54.71 ±8.32	49.11 ±7.57	37.55 ±6.65	1715.43 ±519.13	1561.65 ±427.73	1285.76 ±326.79

In Figure 5.20, the average time in minutes that a UAV takes to handle a given alarm is presented for the different simulation setups. Table 5.2 complements Figure 5.20 presenting the values for the averages plotted in the figure, followed by plus-minus the corresponding value of the standard deviation for the achieved results. Notice in Figures 5.20b and 5.20c that the results for the optimum and flooding-based solutions are presented together as they have similar values, and thus they were merged into a single entity. This is understandable as the flooding solution spreads the alarm messages in all directions, thus reaching the same UAVs as the optimum solution, i.e. the UAVs closer to the location where the alarm was issued. On the other hand, in the pheromone-based solution the alarm messages sent alarmAgent which search for and follow the trails left by the UAVs, as explained in Section 5.3. An additional optimal solution is introduced, the Optimal-T, which minimizes the time-to-handle metric by selecting the closest UAVs from the alarm issuer node that are idle or that have the smaller queue of alarms to handle. Analyzing the proposed pheromone-based solution, an increasing number of appearing threats increases the average time to handle them, which is explainable by the fact that there are situations in which the UAVs may be engaged in handling a previous alarm when they receive a new alarm. This type of situation affected more the setups with 2 UAVs than the setups with 4 UAVs, when they are compared to the respective optimum reference solutions, as it is possible to observe in Figures 5.20b and 5.20c. This is explainable by queuing theory which models the tasks' waiting time, or the delay, to be processed by servers (GROSS; HARRIS, 1998). Here the UAVs can be seen as the servers and the threats as the tasks waiting to be processed, while the time-to-handle metric is the delay to process a task. Notice that for the setup with just 1 UAV, Figure 5.20a, the pheromone-based solution has the same results as the reference solutions. This is explainable because with just 1 UAV in the scenario, there is no alternative to choose a different UAV, i.e. there is just one option. Moreover, the communication delay to alarm delivery (order of milliseconds) is not significant compared to the time required for the physical displacement of a UAV to the position

where the threat was detected (order of minutes). Hence, even the alarm delivery performed by the optimum solutions being more efficient than the one performed by the proposed heuristic, the difference in terms of time is negligible. As a result, in Figure 20a all achieved results for the setup with 1 UAV are merged in a single plot.

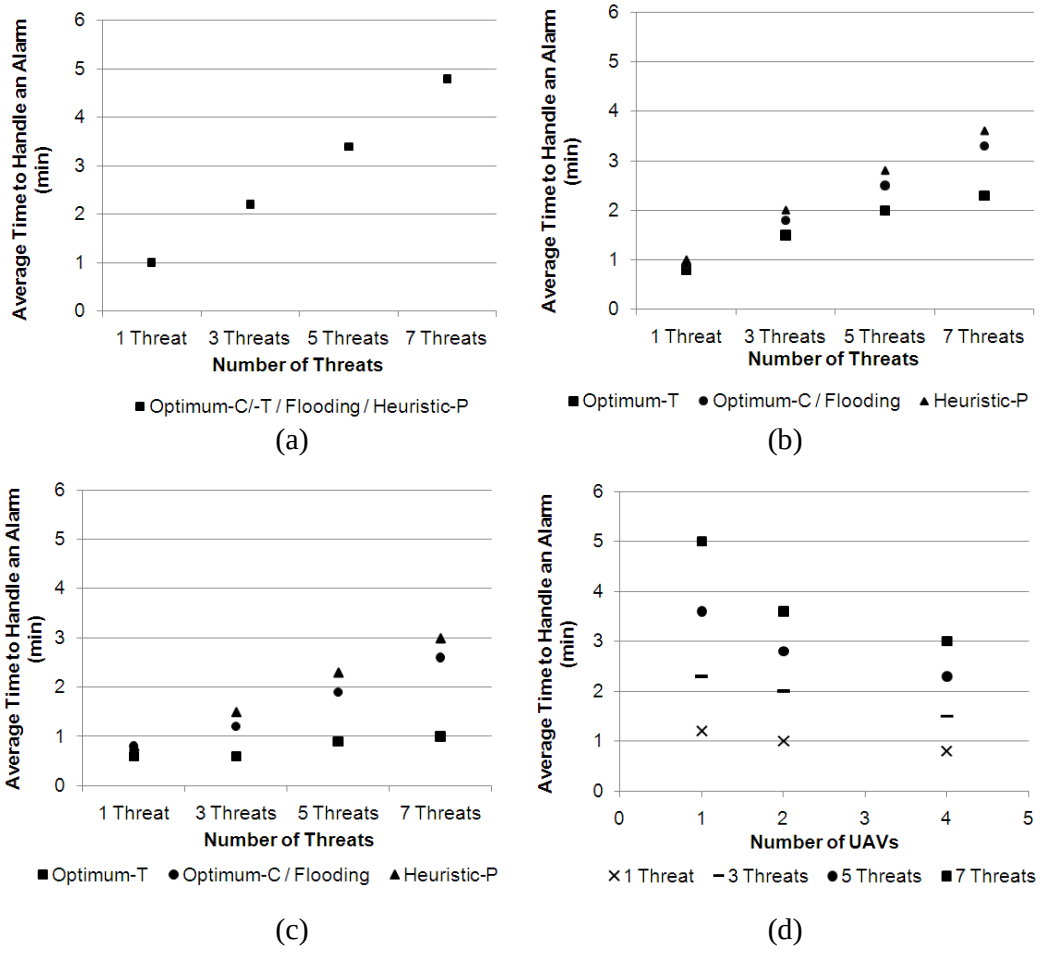


Figure 5.20: Assessment of the "time-to-handle-alarm" metric, in minutes, for the setups with: a) 1 UAV; b) 2 UAVs; c) 4 UAVs; d) comparison according the number of UAVs.

Table 5.2: Average and standard deviation values for the time-to-handle-alarm" metric.

Number of Threats	Heuristic-P			Optimum-T		Flooding	
	1 UAV	2 UAVs	4 UAVs	2 UAVs	4 UAVs	2 UAVs	4 UAVs
1 Threat	1.13 ±0.425	1.011 ±0.268	0.833 ±0.259	0.823 ±0.247	0.611 ±0.222	0.912 ±0.253	0.809 ±0.247
3 Threats	2.301 ±0.633	2.054 ±0.457	1.536 ±0.412	1.536 ±0.406	0.637 ±0.355	1.892 ±0.422	1.223 ±0.382
5 Threats	3.423 ±0.841	2.822 ±0.562	2.355 ±0.455	2.045 ±0.477	0.922 ±0.378	2.572 ±0.543	1.911 ±0.423
7 Threats	4.819 ±0.911	3.647 ±0.737	3.012 ±0.573	2.341 ±0.489	1.009 ±0.401	3.355 ±0.692	2.639 ±0.521

5.5.3 Analysis on Scalability of the Basic Pheromone Mechanism

To assess the scalability of the proposed solution, a second set of simulations was performed for each number of UAVs (1, 2 and 4), in which areas with increasing sizes were considered in addition to the one used as scenario for the first set (4km × 4km), namely: 2Km × 2Km, 6Km × 6Km, 8Km × 8Km and 10Km × 10Km. The assessment was done for the same number of threats (1, 3, 5 and 7) and following the same arrival process described above. The presented results are averages for the group of 100 simulation runs, as the ones presented before for the first simulation set.

The scalability analysis is based on the cost metric. In order to better evaluate the proposed method, the results for this metric were divided according to the specific parts of the overall mechanism (trail-search and trail-follow). First the cost related to the search for trails is presented, i.e. the trail-search, and then the cost for the second part, the alarm forwarding inside a trail, i.e. the trail-follow. To perform a better analysis of the experimental results, the cost metric was normalized as the cost per threat instead of the total cost per setup, as presented in previous simulation set.

Before starting to evaluate the acquired experimental results, some hypothesis can be drawn by observing the way the two parts of the proposed mechanism work and the setup of this simulation set, as follows:

Hypothesis 1: The cost associated with the trail-search should be higher for the larger areas than for smaller ones. This statement is based on the fact that the percentage of the area covered by trails with a given dimension is bigger in the smaller areas than it is in the larger ones.

Hypothesis 2: An increase in the number of UAVs should imply a decrease in the cost associated to the trail-search. The reason for this assumption is that with more UAVs, more trails are available in a given area, increasing the percentage of the area that is covered by them and increasing the probability of an alarm finding a trail.

Hypothesis 3: The cost associated with the trail-search mechanism should be independent of the number of threats. As the results are presented per threat, the number of messages consumed by the trail-search mechanism should not increase with the number of threats.

Hypothesis 4: The cost associated to the trail-follow mechanism should be independent of the number of UAVs and of the number of threats. In principle, these two factors have no direct association with the size of the trails, which is the factor that determines the expected cost for a given trail and can be estimated by the number of sensor nodes that are contained inside its bounds, thus supporting this statement.

Figures 5.21 to 5.23 present the results achieved for the metric cost per threat due to the trail-search mechanism for the different number of UAVs. The numbers above the bars provide the average value followed by plus-minus the value of the standard deviation. All charts in these figures confirm the first hypothesis related to the increased number of messages consumed by the trail-search mechanism according to the size of the area. The second hypothesis is also confirmed, as for all numbers of threats and areas, higher numbers of UAVs imply a smaller number of messages in average. However, the third hypothesis is not confirmed. Observing each figure individually and comparing the results achieved for each number of threats, an interesting finding is made.

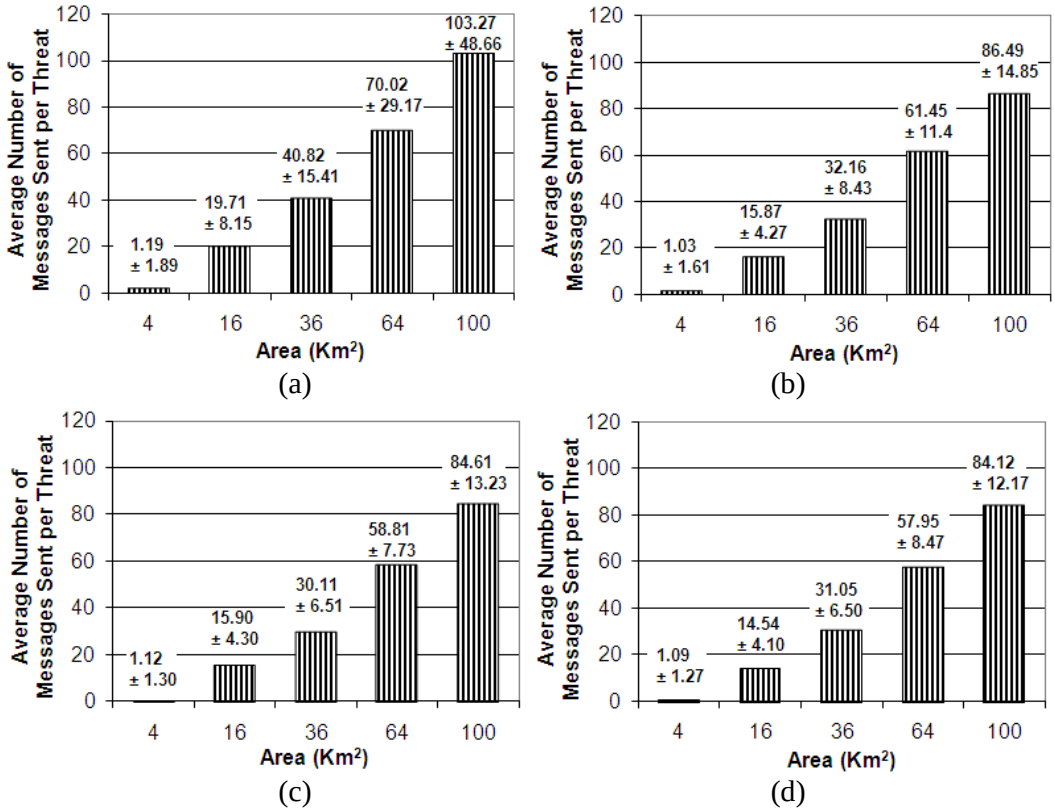


Figure 5.21: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-search mechanism for the simulations with 1 UAV: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

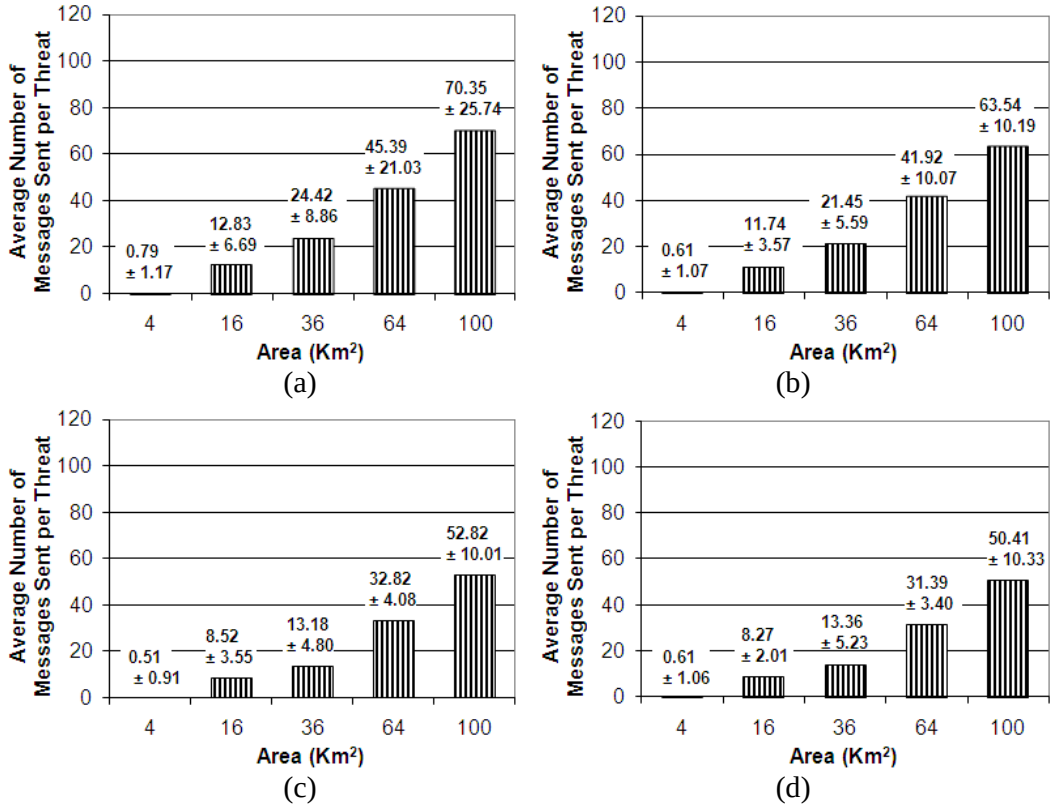


Figure 5.22: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-search mechanism for the simulations with 2 UAVs: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

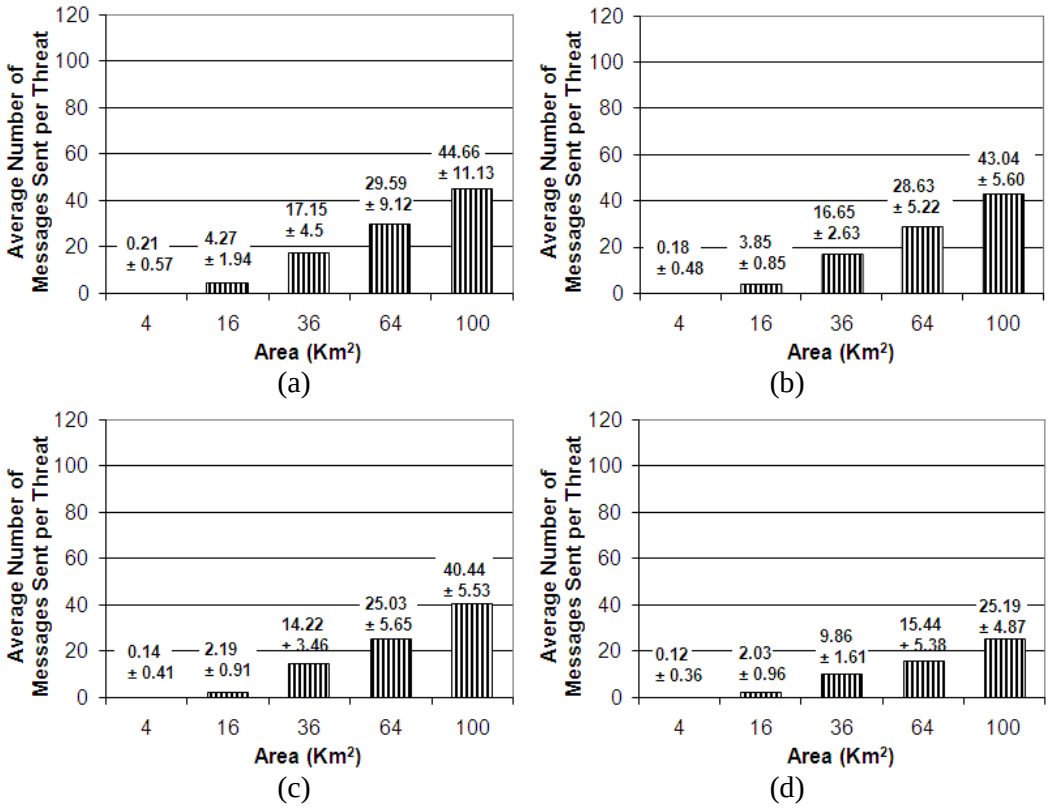


Figure 5.23: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-search mechanism for the simulations with 4 UAVs: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

Analyzing Figure 5.21, it is possible to observe a decrease in the cost metric from part (a) to part (b), which is maintained in parts (c) and (d). In Figure 5.22, a similar behaviour is noticed, but with a remarkable decrease from part (b) to (c), which is maintained in (d), and finally, in Figure 5.23, a remarkable decrease from part (c) to part (d) can be observed. In the case of Figure 5.21 this denotes the situation when the setups with 1 UAV pass from the variation with just 1 threat to the others with more threats. In the case of Figure 5.22, the transition for the setups with 2 UAVs is from the variation with 3 to more threats, while in Figure 5.23 the transition for the setup with 4 UAVs is from the variation with 5 to the one with 7 threats. This observed effect can be explained by the increased mobility that the increasing number of threats impose to the UAVs in the different setups.

Considering the adopted movement model, the UAVs fly according to a random walk pattern, changing their heading angle after each movement step, as explained in Section 5.4. With the parameters used for the performed simulations, during a trail life time the UAVs may change their movement direction up to 10 times. This fact may create a number of pheromone overlapping areas in a trail, which as a result reduces the percentage of the total area effectively covered by a trail. Figure 5.24a depicts an example of trail with overlapping regions (in gray colour), while Figure 5.24b shows another trail without overlap. Notice that for routes of the same size, overlapping

regions are counted twice. This fact reduces the amount of sensor nodes covered by trails with overlapping regions compared to those without overlap.

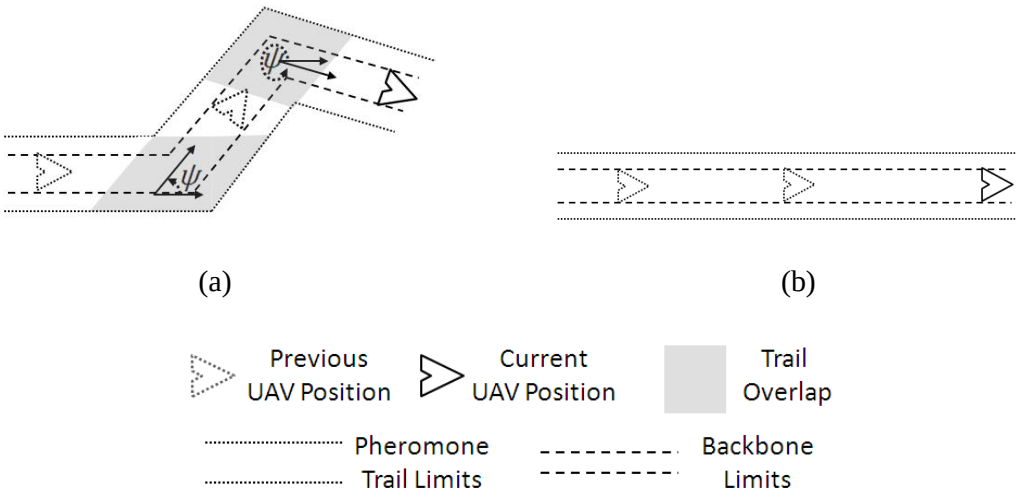


Figure 5.24: Area covered by a trail: a) with overlap; b) without overlap.

When UAVs receive an alarm and are requested to handle a threat, they perform a straight line movement from their current positions to the position in which the threat was detected. This movement is like the one presented in Figure 5.24b, thus creating trails without overlap, increasing the percentage of the area covered by a pheromone trail. In the setups with 1 UAV, for the variations with 1 threat the UAV will be performing its random movement, following a movement such as the one exemplified in Figure 5.24a. However, for the variations with more threats, after having received the first alarm the UAV will fly straight to the threat, as the example in Figure 5.24b, thus increasing the pheromone trail size, which increases the probability of an alarm finding a trail while performing a trail-search. This fact reduces, in average, the number of messages used by this mechanism.

In many cases, new alarms informing about other, and thus new, threats will arrive within, or close to, the time window in which a UAV is already moving towards the most current threat to handle it. In this case, the trail will not present overlaps, or less overlaps compared to the trail left by a UAV performing the ordinary random-walk. For the setups with 2 and 4 UAVs a similar reasoning can be performed, but for a number of threats that exceeds the number of UAVs. This means that the UAVs will be receiving new alarmAgents while going to handle or just after handling a previous one. In these cases, as they have moved straight, their trails have no or less overlaps and are then easier to find by the trail-search mechanism. Notice that for these two setups, with 2 and 4 UAVs, the number of threats immediately higher than the number of UAVs, respectively 3 and 5, do not stress much this observed behaviour, but this becomes clearer in the values 5 and 7. This is explained by the fact that with just one threat more than the number of UAVs, the average results do not change much, but when the

difference is greater, the observed difference appears clearly. This is because it becomes more probable that the UAVs will have pending alarm in queue to be handled.

Figures 5.25 - 5.27 present the results achieved for the metric cost per threat due to the trail-follow mechanism. It is possible to observe that the number of UAVs indeed do not influence in average the number of messages used by the trail-follow mechanism, which partially confirms the fourth hypothesis. However, observing in each figure the variations according to the different numbers of threats, the same changing points observed and discussed for the trail-search results in Figures 5.21 - 5.23 are perceived for each amount of UAVs. However, instead of a decrease in the number of messages sent, as it was observed in the trail-search results, an increase in the number of messages sent due to the trail-follow mechanism is assessed. The reason for this fact is the same as explained above, i.e. the increased size of the pheromone trail in the setup variations in which the number of threats exceeds the number of UAVs. With bigger trails, alarms forwarded by the trail-search mechanism can reach them in portions that are farther from the UAVs, if compared to shorter trails, thus increasing the average of the obtained results of the trail-follow.

An interesting observation to mention is in relation to the variance of the results. It is possible to notice that the results for the trail-follow are much more stable than those for the trail-search. The standard deviation presented together with the average values on top of the bars in the figures that present the results show values for the increasing areas much closer in the trail-follow than the ones observed for the trail-search. This means that the trail-search is much more affected by the increase of the area than the trail-follow, which presents a more deterministic behaviour.

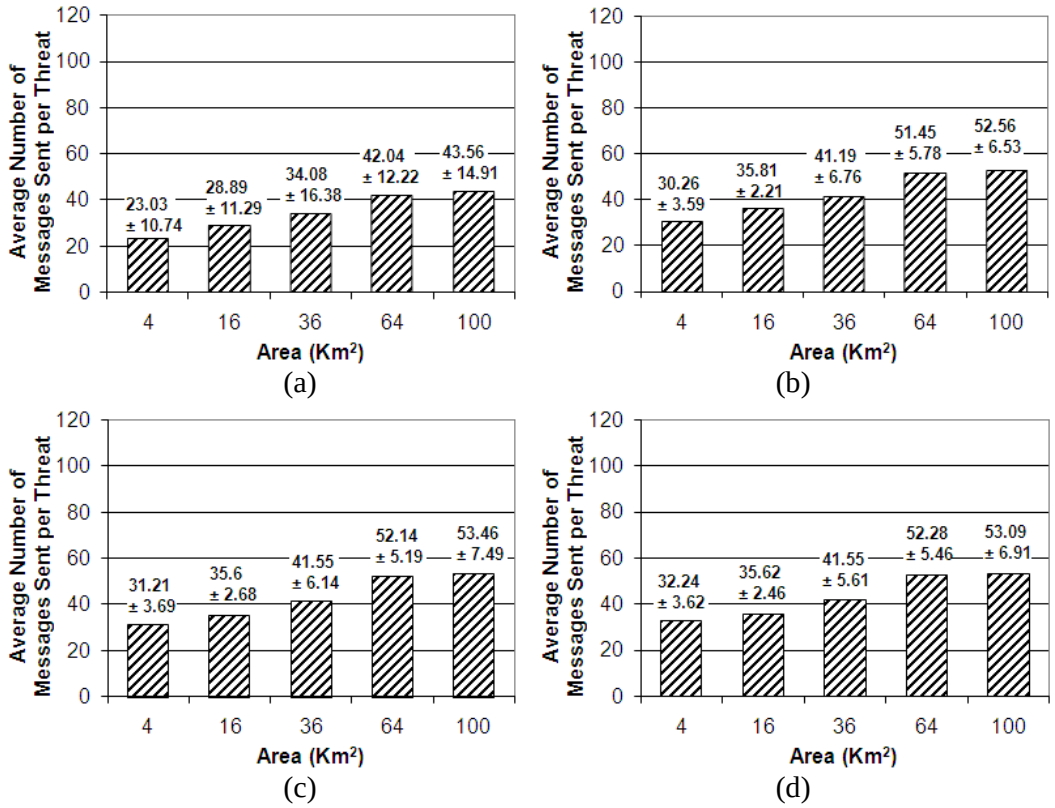


Figure 5.25: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-follow mechanism for the simulations with 1 UAV: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

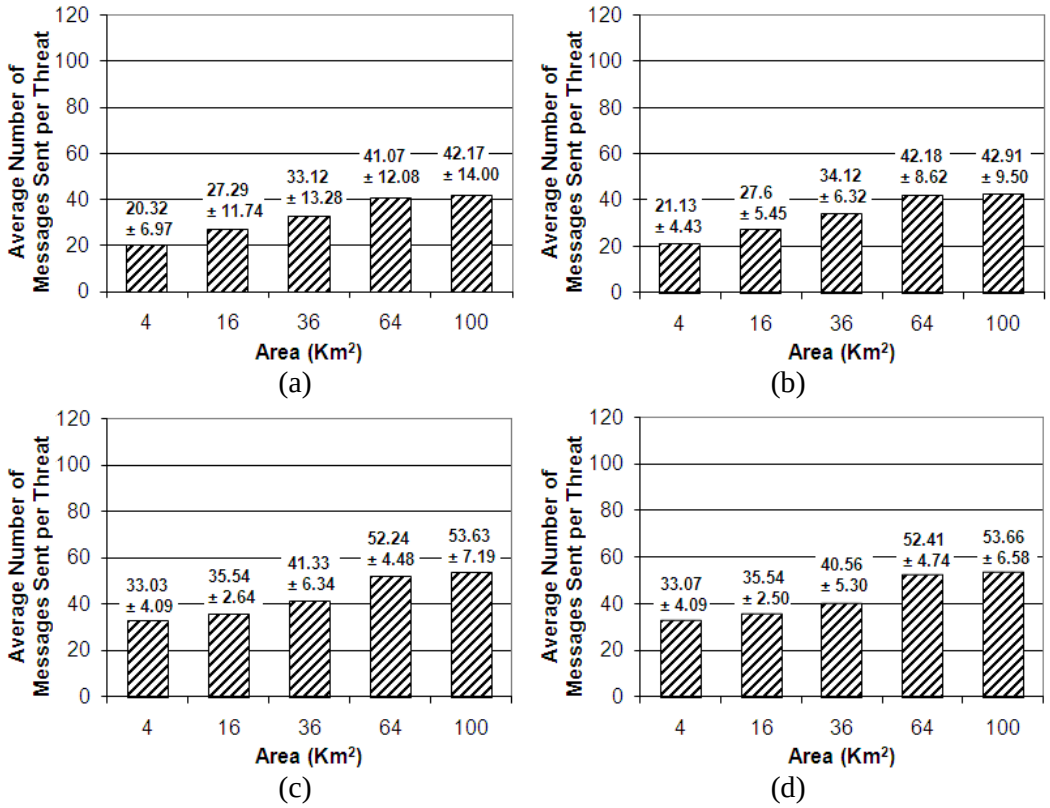


Figure 5.26: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-follow mechanism for the simulations with 2 UAVs: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

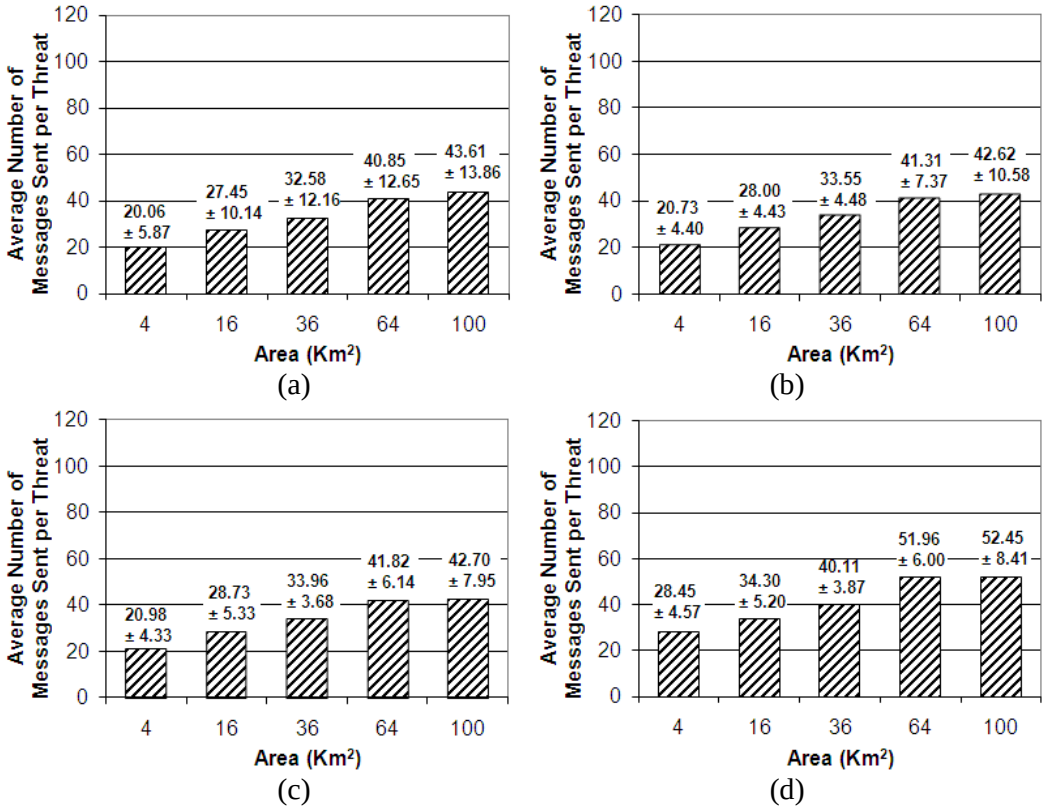


Figure 5.27: Results of the cost metric in terms of the average number of messages sent per threat due to the trail- follow mechanism for the simulations with 4 UAVs: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

To sum up the analysis of this simulation set, Figures 5.28 – 5.30 present the total number of messages in average consumed by the whole mechanism (trail-search plus trail-follow). The upper part of the bars represents the contribution due to the trail-search, while the bottom part represents the contribution due to the trail-follow. These figures allow a direct comparison between the costs associated to each part of the proposed pheromone-based approach.

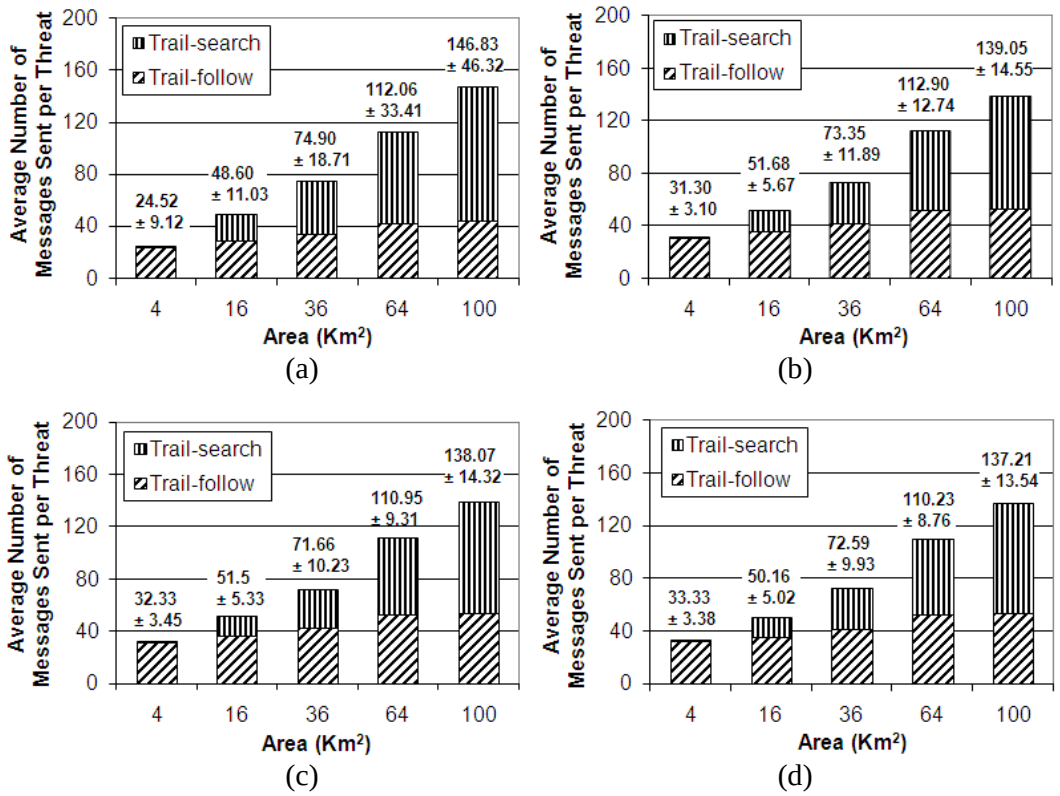


Figure 5.28: Results of the cost metric in terms of the average number of messages sent per threat due to the whole mechanism for the simulations with 1 UAV (the upper part of the bars presents the contribution of the trail-search mechanism while the bottom presents the contribution of the trail-follow): a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

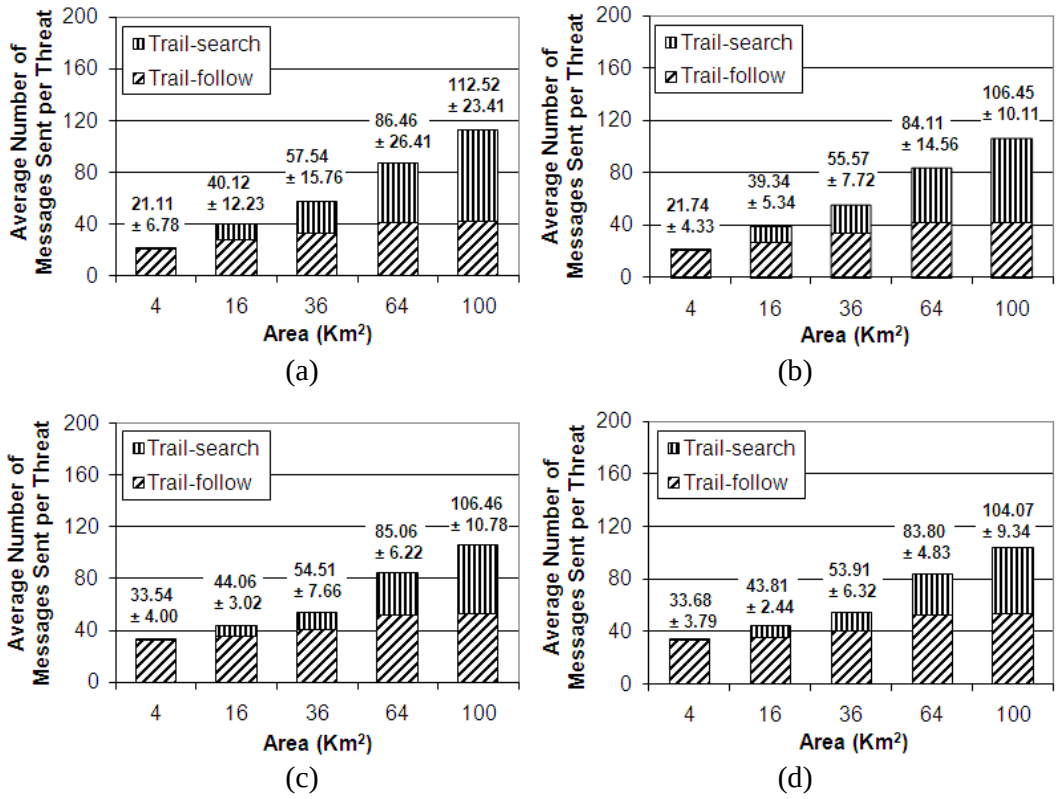


Figure 5.29: Results of the cost metric in terms of the average number of messages sent per threat due to the whole mechanism for the simulations with 2 UAVs (the upper part of the bars presents the contribution of the trail-search mechanism while the bottom presents the contribution of the trail-follow): a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

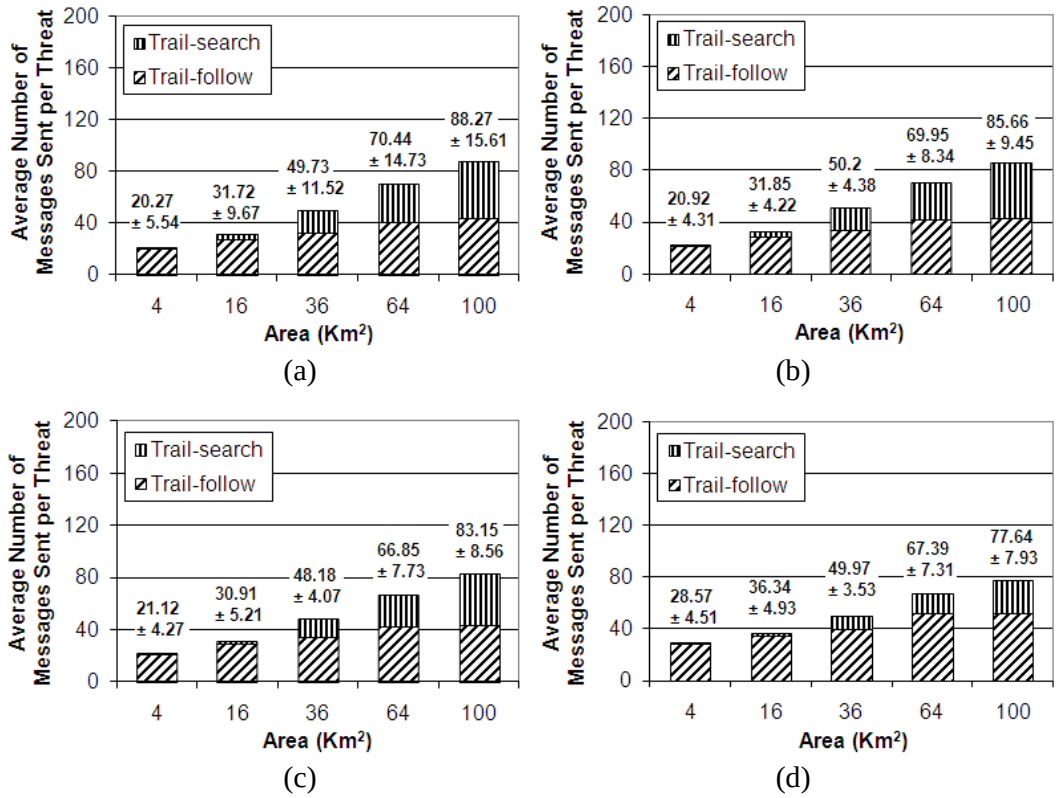


Figure 5.30: Results of the cost metric in terms of the average number of messages sent per threat due to the whole mechanism for the simulations with 4 UAVs (the upper part of the bars presents the contribution of the trail-search mechanism while the bottom presents the contribution of the trail-follow): a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

5.5.4 Influence of the Mobile Node's Movement Pattern

From the analysis of the previous simulation set, a conclusion can be drawn in relation to the observed results: the movement pattern of the mobile nodes has a direct influence in the cost of the proposed pheromone-based alarm delivery. This statement is based on the observed differences in the results both of the trail-search and trail-follow, due to the increase in the number of threats. This fact changed the movement pattern from the random-walk with a number of changes in the movement direction, to a random-walk with longer straight line segments. This finding motivated the execution of a third simulation set. In this new set, the previous results using random-walk for the UAVs' movement pattern with the above described parameters are compared to results achieved by a similar random-walk pattern, but with longer movement steps. The movement steps of this new set are of the size of the maximum length of the pheromone trails, i.e. 5 Km, instead of the movement step size originally used, i.e. 500 meters. This setup will prevent the UAVs from changing their direction while flying during the lifetime of a trail, unless they reach the edges of the surveillance area or in the case they have to change their direction to move towards a threat just informed by a new alarm that will be handled.

For this simulation set an area of 10 Km × 10 Km is used as scenario, in which 2 UAVs patrol the area and variations with the same number of threats were tested, i.e. 1, 3, 5 and 7 threats. The assessed metric is the same as in the previous simulation set, the cost per threat for both parts of the pheromone-based approach, i.e. trail-search and trail-follow. The hypothesis to be tested is if the number of threats will not influence the cost results, thus confirming the conclusion about the influence due to the movement pattern. Figure 5.31 presents the achieved results for the extended movement step compared to the original one.

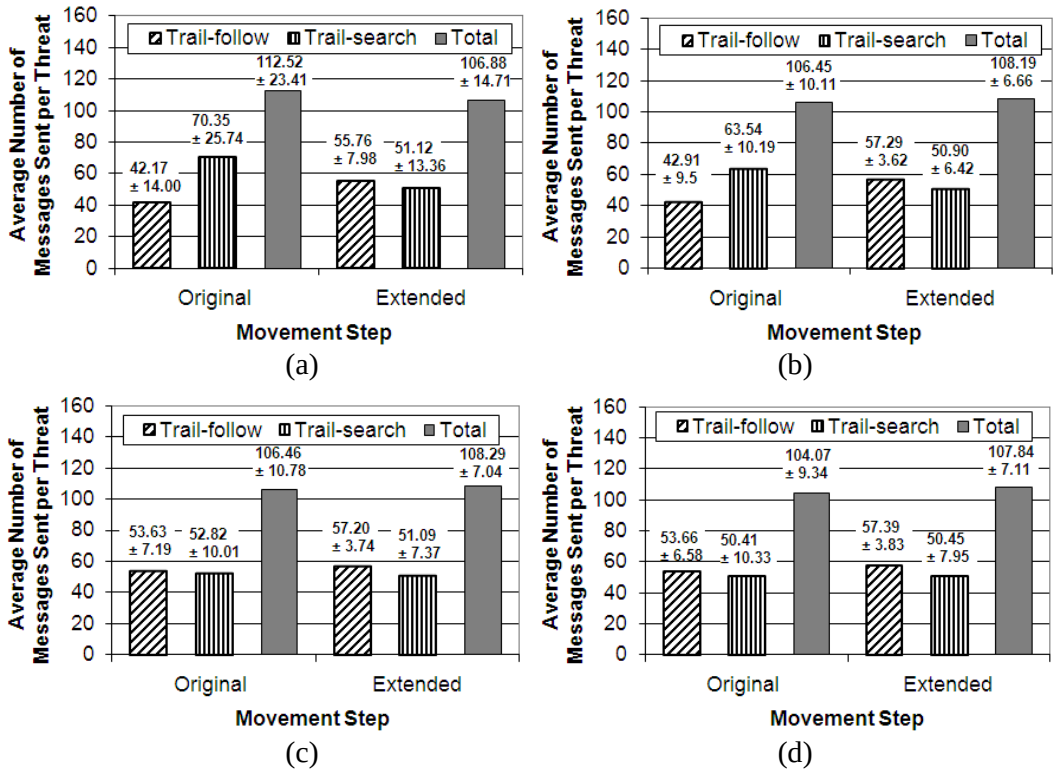


Figure 5.31: Results of the cost metric in terms of the average number of messages sent per threat due to the trail-search, trail-follow and the total (trail-search plus trail-follow) for the simulations with 2 UAVs for both original and extended movement step: a) 1 Threat; b) 3 Threats; c) 5 Threats; d) 7 Threats.

As can be observed in the results achieved with the extended movement step, the number of messages consumed by both parts of the coordination mechanism is almost the same for the different number of threats, within a statistical variation. The point that presented the difference from the simulation variations from 3 to 5 threats that is observed with the original movement step is not observed in the results with the extended step (from Figure 5.31b to Figure 5.31c). This confirms the formulated hypothesis that the number of threats should not influence the results and that in fact the observed differences in the previous simulation set are due to the change in the movement pattern that is created by the threat handling in the simulations with the original movement step. Moreover, a decrease in the values of the standard deviation

can be noticed for both trail-search and trail-follow, which confirms that they became more deterministic in the results for the extended movement step than for the original step. For the total number of messages sent, there is no big difference between the results originally achieved compared to the ones achieved with the extended movement step. In relation to the variability of the results, Figure 5.31 shows that it is smaller with the extended step, as values of the standard deviation that comes together with the values of the averages are smaller for the extended step, which is consistent with the results for the trail-search and trail-follow.

Even after concluding that the real effect of the different number of threats in the results is due to the difference that they imply in the UAVs' movement pattern, we decided to keep the original movement step and the variation of the number of threats for the further simulation sets. This decision is based on the fact that simulation sets with varying step lengths would create a similar effect than varying the sizes of the areas with the same movement step, i.e. vary the expected percentage of the area that can be covered by a trail, which was already tested. Moreover, maintaining the pattern of the simulations makes it easier to compare new results with previous comparable simulation sets. Another aspect that is noteworthy to mention is that the study of different movement patterns is very dependent on specific requirements of particular applications. This is especially important for non-functional requirements, such as security and secrecy, due to the fact that some movement patterns can be easier recognizable by hostile entities than others. In spite of the importance of this subject, a deeper study about it, as found in (BAI; SADAGOPAN; HELMY, 2003) and (COOPER; MEGHANATHAN, 2010), is beyond the goals of this work.

To keep the simulations following the same standard and the presentation of the achieved results following the same format, the original movement pattern is maintained for the further simulations sets.

5.5.5 Assessment of the Advantage in using the Backbone in the Pheromone Trail

Taking into account the distribution of pheromones over the ground sensor nodes presented in Section 5.3.1 and the proposal of using the central part of the trail as a limiting border to the propagation of an alarm that occurs inside a trail, a fourth set of simulations was performed to assess the effectiveness of this strategy. In this fourth set, the basic conditions presented in the first simulation sets were maintained, but the pheromone trail has no backbone, such that the alarm forwarding is performed as illustrated in Figure 5.5a. The performed simulation had a scenario with an area of 10Km $\times$ 10Km, in which 4 UAVs were patrolling the area, and there were the same increasing numbers of threats. Results for cost in terms of messages sent were compared to those of the corresponding setups presented in the previous experiments, in which the pheromone trails have backbone.

From the results in Figure 5.32a, which uses a semi-log scale, it is possible to compare both versions of the pheromone strategy with the reference solutions and also to observe that they have similar behaviour, being much less costly than the flooding based one, and not too far from the optimum solution. A more detailed analysis between them is provided in Figure 5.32b, where a linear scale is adopted. The figure shows that

the usage of the backbone, which was defined with half of the width of the whole trail, provided a reduction of in average 30% in the number of messages sent for each number of threats. These results are consistent with those presented in Section 5.5.3, particularly the ones presented in Figure 5.30, which shows that the trail-follow mechanism contributes with approximately 50% of the amount of messages consumed by the entire mechanism, unless for the variation with 7 threats, in which the trail-follow requires a higher percentage (Figure 5.30d), as was discussed above. By using a backbone layer with half of the width of the trail, it is expected that the number of messages consumed by the trail-follow drops to half of the number of messages required by this mechanism in a trail without backbone. As a result, the total number of messages sent by the entire mechanism (trail-search and trail-follow) is expected to be reduced in average 25%. Thus, considering the case for 7 threats, in which the trail-follow requires relatively more messages, the achieved savings of approximately 30% is a reasonable result. Table 5.3 complements Figure 5.32 providing the average values plotted in figure followed by the plus-minus the value of the corresponding standard deviation.

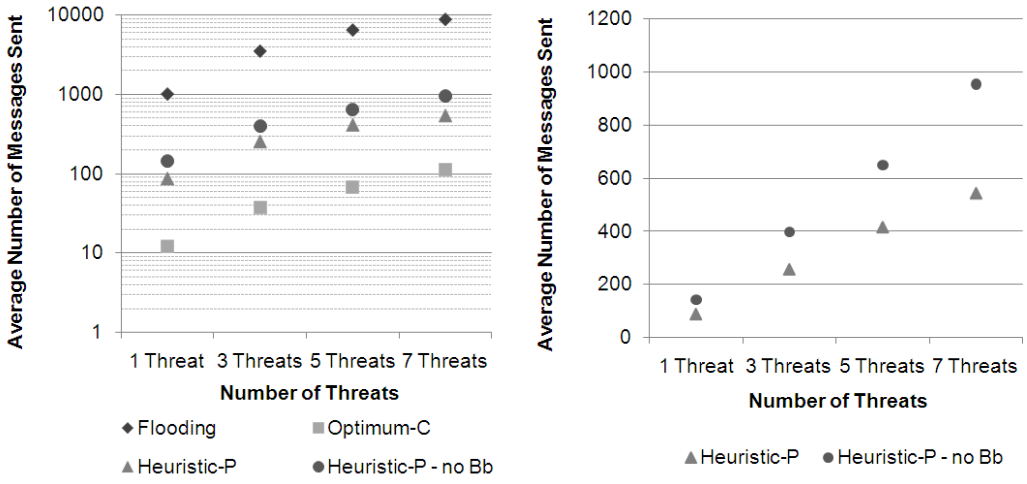


Figure 5.32: Cost of the pheromone strategy with (Heuristic-P) and without (Heuristic-P – no Bb) backbone: a) cost comparison with the two reference solutions in a semi-log graph; b) cost comparison between the variations of the pheromone strategy in a linear graph.

Table 5.3: Average and standard deviation values for the cost in terms of messages sent for the comparison of the heuristic-P with and without backbone.

Number of Threats	Flooding	Optimum-C	Heuristic-P	Heuristic-P-no Bb
1 Threat	1015.54 ±397.65	12.86 ±3.43	88.27 ±15.61	143.32 ±15.47
3 Threats	3523.72 ±986.79	37.13 ±7.07	256.98 ±28.22	396.8 ±41.52
5 Threats	6488.93 ±1532.55	67.59 ±13.43	415.73 ±44.37	648.32 ±67.82
7 Threats	8837.48 ±2167.33	112.04 ±20.74	543.49 ±56.16	953.74 ±94.43

5.5.6 Considering UAVs with different capabilities

It is most likely that a surveillance system uses several UAVs equipped with different types of sensors (suitable to perform reconnaissance tasks of threats of different types). The appearance of threats of different types in the surveyed area is a highly probable situation requiring different types of sensors to perform their reconnaissance tasks. The applicability of the sensor that equips a UAV defines the utility of this UAV to handle a given threat.

To evaluate the proposed approach in this context in which there are threats of different types and UAVs equipped with different sensors, a fifth set of simulation experiments was designed. In this set there are three types of threats and there are UAVs equipped with 3 different types of sensors. To calculate the applicability of a sensor to handle a threat in (5.2), for each type of sensor it is provided a constant value for $\varphi^{j}_{ui}(t)$ in relation to the simulation time, as follows:

$$\varphi^{j}_{u_i}(t) = \varphi^{j}_{u_i} = \begin{cases} 1, & \text{if } j = k \\ 0.5, & \text{if } j \neq k \text{ and } k \in K_j \\ 0, & \text{if } k \notin K_j \end{cases} . \qquad (5.9)$$

where j is the type of sensor and k is the type of threat. The value 1 determines that the sensor is suitable to handle the threat, while the value 0.5 means that the sensor is partially suitable to handle the threat. The value 0 means that the sensor is not suitable to handle the threat.

For degradation in the measurements due to operation conditions ($Op_{j,p\tau}(t)$) used in the computation of (5.2), the values are also constant in relation to the simulation time. In these simulations the operation conditions are exemplified by weather conditions at the position where the threat was detected, informed about in the alarm. In the case of an alarm indicates bad weather, the degradations in the sensors' applicability are

randomly defined according a uniform distribution in three different intervals according to the type of sensor:

$$Op_{j,p}(t) = Op_{j,p} = \begin{cases} random(0,0.33), & if \quad j = 1 \\ random(0.33,0.66), & if \quad j = 2 \\ random(0.66,1.00), & if \quad j = 3 \end{cases}. \qquad (5.10)$$

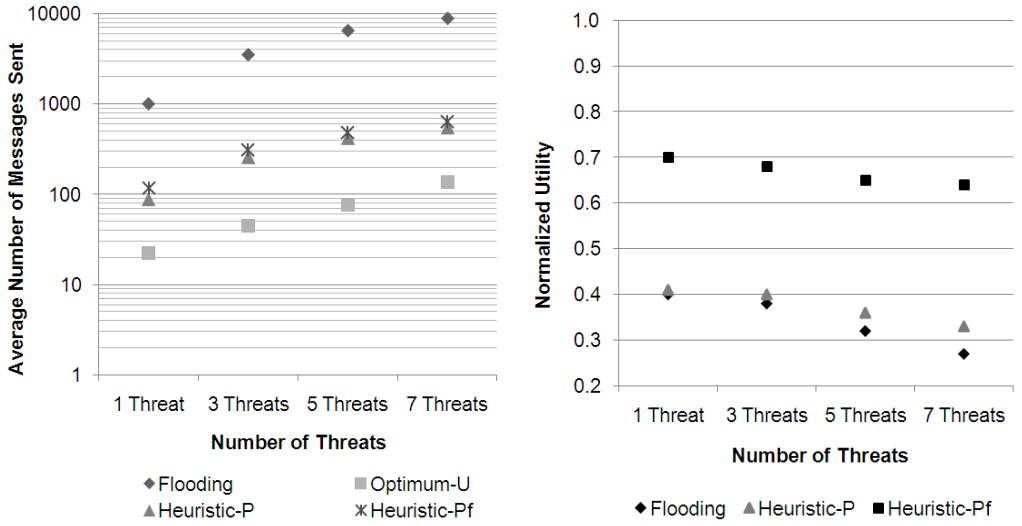
where j is the type of the sensor and the p_τ is the position where the threat was detected. According to defined in (5.10), the sensor type 1 is the most affected by the weather conditions, and its applicability is reduced to up to 33%, while the sensor type 3 is the one that may suffer no degradation. If the alarm does not indicate bad weather, the sensor keeps its normal applicability, i.e. $Op_{j,p_\tau}(t) = 1.00$.

The evaluation of the UAVs equipped with different capabilities requires a third metric, called "utility", which assess the utility in employing a given UAV to handle a given threat. A test scenario with an area of 10Km × 10Km, with 4 UAVs and the same increasing numbers of threats used in previous simulation sets is used. The evaluation of the pure pheromone-based strategy (heuristic-P) was compared to its variation that considers the differences among UAVs, i.e. pheromones with different flavours (heuristic-Pf). For this simulation set, a simplified version for the general model of the UAV states presented in FSM depicted in Figure 5.2 is used. In this implemented model, there is no negotiation among the UAVs, which means that once an alarm is assumed by a UAV, it keeps this alarm until it can handle the respective threat.

The optimum (reference) solution is slightly different in this simulation set. It works in a similar way as described before, using the smallest possible number of messages to deliver an alarm to a UAV, but, instead of selecting the UAV closer to the position in which the threat was detected, it selects the one that has the maximum utility value to handle the threat. This means that it maximizes the utility metric and for this reason is called Optimum-U. The flooding solution could be changed in the same way, i.e. defining as stop criterion the alarm arrival at the UAV with maximum utility, instead of the closest one. However, this modification would make the flooding similar to the optimum solution in terms of the utility metric, leaving just the cost metric as a difference among them. For this reason, it was decided to keep the flooding solution as it was used in the previous simulation sets.

The results presented in Figure 5.33 show that, in terms of cost, heuristic-Pf has a pattern very similar to the one presented by the pure pheromone one (Figure 5.33a), requiring in average 15% more messages. This additional cost is due to the fact that, when an alarm is issued in the system using heuristic-Pf, the alarm does not simply takes the first trail that it finds (if it was not issued inside a trail). Instead, it searches for a trail of a pheromone with the flavour that indicates the UAV equipped with the sensor that is suitable (or at least partially suitable) to handle the threat that triggered the alarm while it is performing the trail-search. On the other hand, in the pure pheromone strategy, in which the differences among UAVs are not considered, an alarm takes the first trail that it encounters while performing the trail-search. Table 5.4 complements

Figure 5.33a presenting the values for the averages plotted in figure followed plus-minus the corresponding values of the standard deviation. Observing the utility results (Figure 5.33b), it is possible to see the poor values achieved by both heuristic-P and the flooding, if compared to the heuristic-Pf. This is explainable by the fact that both flooding and heuristic-P do not take into account the UAVs' capabilities to handle a threat. It may happen several times that they select UAVs that do not even have a sensor that is useful to handle a given threat, so this selection provides zero utility value. On the other hand, heuristic-Pf will select trails with a pheromone flavour of UAVs that are able to handle a given threat. Complementing Figure 5.33b, Table 5.5 presents the values for the averages plotted in the figure, followed by plus-minus the corresponding values of the standard deviation of the achieved results.



(a) (b)

Figure 5.33: Results for the pheromone strategy taking into account the differences among UAVs Heuristic-Pf compared to the pure pheromone one Heuristic-P: a) cost metric in number of messages; b) normalized utility in relation to Optimum-U.

Table 5.4: Average and standard deviation values for the cost in terms of messages sent assessed for the flooding, optimum, heuristic-P and heuristic-Pf.

Number of Threats	Flooding	Optimum-U	Heuristic-P	Heuristic-Pf
1 Threat	1015.54 ± 397.65	22.32 ± 5.67	88.27 ± 15.61	116.45 ± 23.54
3 Threats	3523.72 ± 986.79	45.25 ± 9.91	256.98 ± 28.22	309.63 ± 36.66
5 Threats	6488.93 ± 1532.55	75.87 ± 15.75	415.73 ± 44.37	488.13 ± 53.23
7 Threats	8837.48 ± 2167.33	136.35 ± 27.13	543.49 ± 56.16	633.27 ± 69.43

Table 5.5: Average and standard deviation values for the utility metric assessed for the flooding-based solution, heuristic-P and heuristic-Pf.

Number of Threats	Flooding	Heuristic-P	Heuristic-Pf
1 Threat	0.403 ±0.157	0.412 ±0.167	0.731 ±0.082
3 Threats	0.385 ±0.158	0.405 ±0.178	0.686 ±0.088
5 Threats	0.321 ±0.132	0.364 ±0.142	0.655 ±0.085
7 Threats	0.272 ±0.127	0.335 ±0.147	0.642 ±0.084

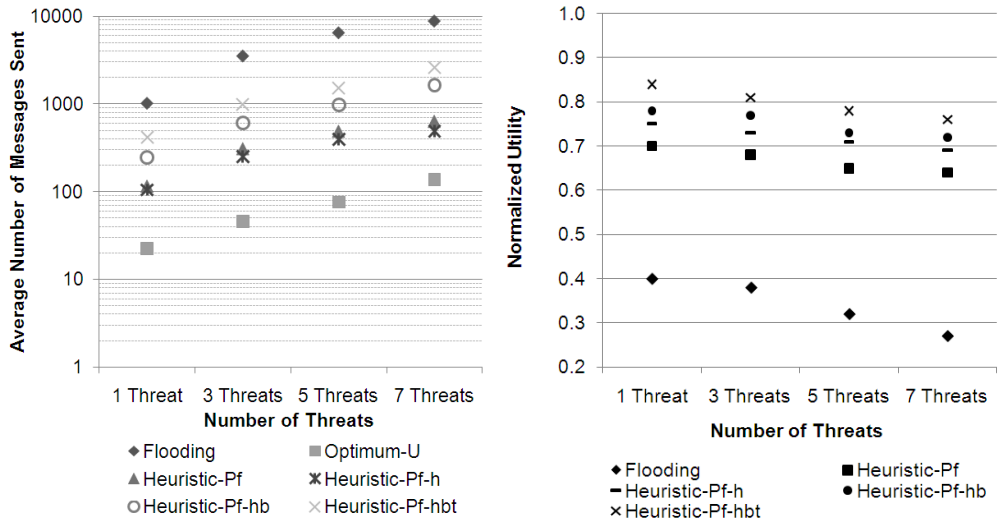
5.5.7 Improving Utility Results

Aiming at improving the utility results achieved by the pheromone-based cooperation strategy, three enhancements were proposed in Section 5.3.2. To analyze how effective these enhancements are to achieve this goal, a sixth simulation set was defined. In this set each variation is compared to the previously analyzed heuristic-Pf, as well as to the reference optimum and flooding solutions. The cost metric for the three enhanced mechanisms includes the cost in transmitting the pheromone information among the ground sensor nodes to spread the trails, as described in Section 5.3.2.

It can be observed in Figure 5.34a an increased cost associated with the dissemination of the pheromone information by the ground sensor nodes. Notice that the first variation, heuristic-Pf-h, does not present an increase in the cost in relation to heuristic-Pf, but even a small decrease. This is understandable, as in this variation the spread of the pheromones information is done exclusively by the hitchhiking mechanism, without any dissemination among the ground sensor nodes, thus incurring in no additional costs. Moreover, this dissemination performed by the hitchhiking mechanism makes it easier to the alarmAgent find a trail of a suitable UAV during the trail-follow, thus diminishing the communication cost. On the other hand, the two other variations, heuristic-Pf-hb and heuristic-Pf-hbt, present a significant increase in the cost due to the additional messages used to spread the pheromone information. However, it is noteworthy that even with this increase they are far from the flooding cost, having less than one third of that cost. Table 5.6 complements the information plotted in the graph of Figure 5.34a, presenting the averages values for the cost metric followed by plus-minus the value the corresponding standard deviation.

Observing the utility metric in Figure 5.34b, an increase in the utility of about 5.5% can be achieved from heuristic-Pf to the heuristic-Pf-h variation. From this one to the next variation, heuristic-Pf-hb, the increase is a little bit smaller, less than 3%. From the results achieved by heuristic-Pf-hb to the ones achieved by heuristic-Pf-hbt, the increase is higher, around 5.25%. These results show that the higher increases in utility are due more to the hitchhiking mechanism than to the dissemination of the pheromone information by the ground sensor nodes among them. Even if this dissemination indeed presents an enhancement in the utility results, if analyzed together with the cost, it is debatable if the gains in terms of utility outweigh the costs in using this alternative.

Figure 5.34b is complemented by Table 5.7, which presents the plotted average values followed by the corresponding standard deviation values.



(a) (b)

Figure 5.34: Results achieved with the variations in the pheromone dissemination: a) cost metric; b) normalized utility.

Table 5.6: Average and standard deviation values for the cost in terms of messages sent assessed for the flooding, optimum and the different variations of the heuristic-Pf.

Number of Threats	Flooding	Optimum-U	Heuristic-Pf	Heuristic-Pf-h	Heuristic-Pf-hb	Heuristic-Pf-hbt
1 Threat	1015.54 ± 397.65	22.32 ± 5.67	116.45 ± 23.54	104.74 ± 18.33	245.43 ± 42.31	413.43 ± 66.74
3 Threats	3523.72 ± 986.79	45.25 ± 9.91	309.63 ± 36.66	252.52 ± 28.45	602.75 ± 71.23	988.19 ± 95.86
5 Threats	6488.93 ± 1532.55	75.87 ± 15.75	488.13 ± 53.23	395.78 ± 41.87	978.11 ± 106.77	1520.56 ± 143.83
7 Threats	8837.48 ± 2167.33	136.35 ± 27.13	633.27 ± 69.43	488.11 ± 52.96	1653.78 ± 178.45	2606.41 ± 239.57

Table 5.7: Average and standard deviation values for the utility metric assessed for the flooding-based solution and the different variations of the heuristic-Pf.

Number of Threats	Flooding	Heuristic-Pf	Heuristic-Pf-h	Heuristic-Pf-hb	Heuristic-Pf-hbt
1 Threat	0.403 ±0.157	0.731 ±0.082	0.75 ±0.085	0.78 ±0.0808	0.84 ±0.078
3 Threats	0.385 ±0.158	0.686 ±0.088	0.73 ±0.078	0.77 ±0.059	0.81 ±0.091
5 Threats	0.321 ±0.132	0.655 ±0.085	0.71 ±0.079	0.73 ±0.083	0.78 ±0.086
7 Threats	0.272 ±0.127	0.642 ±0.084	0.69 ±0.079	0.72 ±0.076	0.76 ±0.093

5.6 Summary

The proposal to support cooperation among static and mobile sensor nodes presented in this chapter explores a bio-inspired solution based on the stigmergy concept. The idea is to distribute information about the movement of the mobile sensors over the static ones, mimicking the spreading of pheromones performed by ants in their habitat. Simulations where used to evaluate the performance of the proposed approach and its variations, assessing the cost associated with the communication among the sensor nodes. Compared to a flooding based reference solution, the results achieved with the proposed approach were one order of magnitude lower than those achieved by flooding, i.e. order of thousands for the flooding while hundreds for the proposed approach. Moreover, for the cases in which mobile sensor nodes with different capabilities are considered, results related to the utility in engaging a given mobile sensor node to respond a given alarm were also assessed. This assessment took into account the different variations of the basic pheromone mechanism. The achieved results showed an enhancement of in average 10% from the heuristic-Pf to the heuristic-Pf-hbt variation in terms of the utility metric. However, this better result for the utility lead to an increase in the associated costs due to communication of approximately 4 times in average, which states a trade-off between these two parameters, utility and cost.

6 IMPLEMENTATION ASPECTS OF THE PERFORMED SIMULATIONS

As mentioned in Section 1.7, the performed experiments to test the developed approaches presented in Chapters 3, 4 and 5 were conducted by means of simulations using GrubiX simulation tool. The goal of this chapter is to provide an overview of the GrubiX simulator, its development framework, and how the software agents used in the performed simulations were implemented using the resources provided by the GrubiX framework.

6.1 GrubiX Simulator

GrubiX (HEIMFARTH; FREITAS, 2011) is an Open Source tool developed to support the simulation of ad hoc networks and is an evolution of the ShoX simulator project (LESSMANN; HEIMFARTH; JANACIK, 2008). It is object-oriented and implemented in the Java programming language, providing an easy way to extend its framework by using class inheritance and interface implementation. The simulator consists of a Java project that can be loaded in any IDE, such as Eclipse (OBJECT TECHNOLOGY INTERNATIONAL, 2003), in which each a new simulation is a new project that has a dependence link to the GrubiX project. To implement a new simulation, the user just need to extend the classes defined in the GrubiX project according to the protocols and models that he/she wants to develop.

GrubiX provides a flexible configuration method, which consists of a XML configuration file template. For each project this configuration file is used to specify the simulation parameters, such as the dimensions of the simulated environment, the number of nodes, the nodes' types, as well as the protocols used in each of their network layers. The simulator provides the flexibility to use combinations of protocols in the different layers, in which it is not obligatory to choose a specific protocol for all layers. This means that the user may define the protocol for just one layer, and the tool fills the other layer with "dummy" debugging protocols that just wrap or unwrap packets from the upper and lower layers and pass to them the respective resulting packet accordingly. This is an interesting feature which simplifies the debugging as it is possible to analyse the behaviour of the protocols for each layer separately. Moreover, this added flexibility also provides means to partition the layers' functionalities in innovative ways or to design cross-layered approaches, according to the user needs.

The output of the simulations is stored in one log and one report file. The log file stores the evolution of the simulation while the report file stores statistics results of interest defined by the user. To analyse the output of a simulation, the simulator provides a visualisation tool for the log file that shows a two dimensional perspective of the nodes' interaction in which the different elements of the network are graphically represented. The visualisation tool presents the log in a movie like style, in which the simulation evolves according to the performed simulation steps. What a simulation step shall correspond to in terms of absolute time steps is specified in the XML configuration file. Figure 6.1 presents a screenshot of its graphical user interface (GUI) in which small circles are used to represent nodes distributed in a gray squared area. The nodes are connected by lines, which represent the communication links among them, and they are surrounded by a gradient coloured circle, which represents their sensing range. The red coloured small squares represent the packets being sent by the blue node to its neighbour nodes. On the bottom part of the GUI there are commands to run the visualisation allowing the simulation to move forward or backward in time, and to increase and decrease the presentation speed. On the right hand side of the GUI there are some options that can be configured. One of them is the possibility to show or to hide the lines representing the communications links among the nodes (No Graph / With Graph). It is also possible to change the options to present the sensing range (No Sensor Range / Normal Sensor Range / Gradient Sensor Range) and presentation of the communication packets, that can appear as small squares (Packet Normal) sent over links or as an animated increasing circle around the sender node (Packet Wave). The Packet Wave option illustrates radio packages sent out to the surrounding neighbourhood. It is also possible to set the simulation to run in real-time mode, in which the speed is adjusted to display the visualisation in a speed that matches with the absolute time instead of the simulation steps as it chooses by default. This is useful to follow real world processes that evolve in a rather slow fashion compared to the network events, such as mobile sensor nodes motion. Further it is possible to configure the tool to present the identification number of the nodes (NodeID) as labels above the nodes. The size of the circles that represent the nodes is also configurable, by choosing the number of pixels to be displayed to represent these circles.

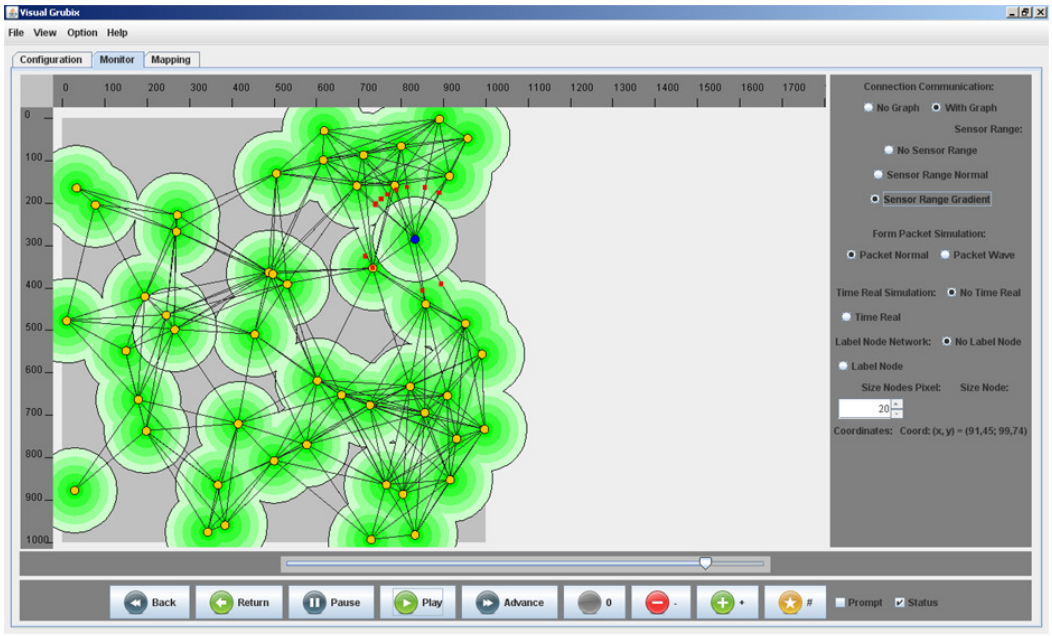


Figure 6.1: Visual GrubiX – Visualisation tool of the GrubiX ad hoc network simulator.

GrubiX is a discrete event simulator in which the events are ordered according their occurrence and then processed accordingly. This works as a schedule for coming events in which the simulator when it has processed an event and a method to update a clock (keeping track of the simulated current time), it takes the next event in queue and executes its associated method.

The method associated with an event executes its logic, possibly generating new events. To do this it can call other methods. The occurrence of events affects the simulation state, requiring the update of counters and statistics used to generate log and report files.

Figure 6.2 presents the class diagram with structure of the events in GrubiX.

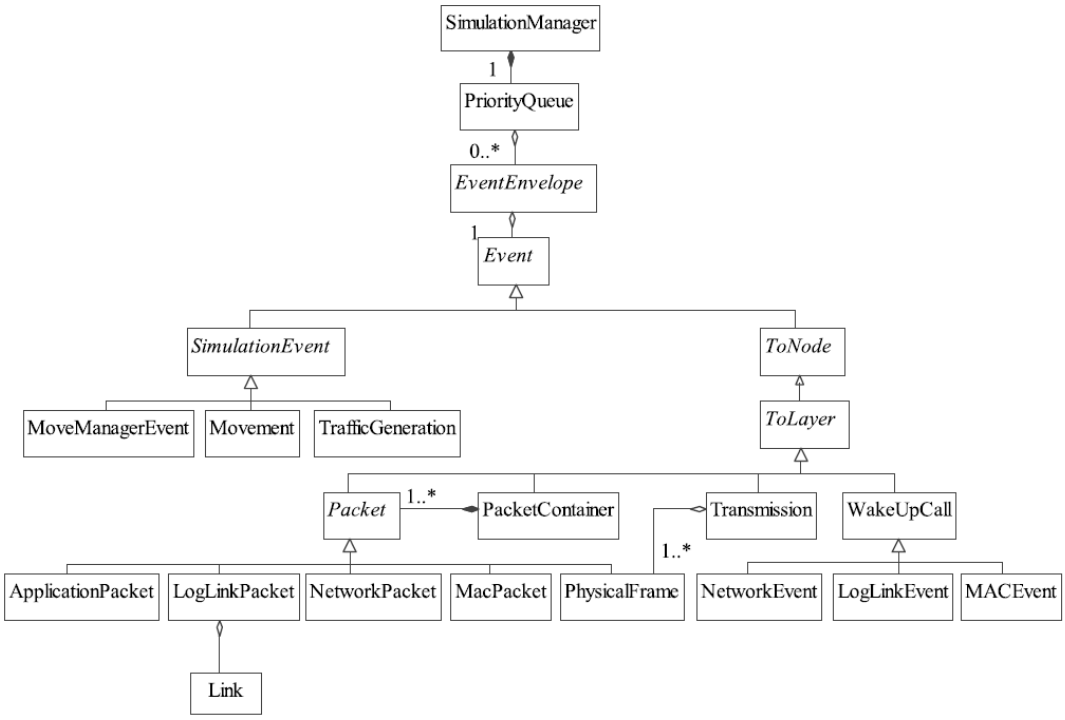


Figure 6.2: Class Diagram for the structure of events in GrubiX.

On the top of Figure 6.2 is the class `SimulationManager`, which is the main class of the simulation framework. This class controls the whole simulation flow and have access to all global objects used by the simulator. The events control is performed in this class, which contains a reference to event queue (`PriorityQueue`).

The `PriorityQueue` is responsible for all the events that have to be executed in the system. The events are stored in the queue by using another class called `EventEnvelop`. This class store the event information and a method to process the respective encapsulated events. The class `Event` is an abstract superclass that represents all simulated events. The events can be of two types: simulation events (`SimulationEvent`) and events related to the nodes' communication protocol stack (`ToNode`).

The simulation events are used by the simulator to model events that are not directly related to the nodes' communication protocol stack, but are related to other aspects such as the nodes' movement or generation of traffic in the network.

The events related to the nodes' communication protocol stack can affect all layers in the stack, being modelled by the class `ToNode`, or affect specific layers, hence being modelled by the class `ToLayer`. Events of the type `ToLayer` are dispatches to a certain layer in the protocol stack, of which the most important is the class `Packet`. Logically, packets from a layer in one node are addressed to the same layer in another node, but to reach the physical layer to be transmitted, they have to pass through the layers bellow its original one in the sender node. Packets of upper layers are encapsulated by packets in the layer bellow. At the receiver side the process is done in inverse order, and the packets are thus delivered to the upper layers until they reach the

destination of a communication. This follows the architecture and operation of layered (stacked) communication protocols.

The `WakeUpCall` is an important event, which is used to schedule future events in the simulator. An example of its usage is to implement sending of periodic packets, such as the pheromone beacons sent by the UAVs to ground sensor nodes presented in Chapter 5.

Besides the structure of the classes related to the handling of events, a second important class diagram is the nodes' structure, Figure 6.3. The main class is the Node, which models the nodes simulated. A node is specified by its network stack layers. The classes that model the layers inherit class information from a common class `Layer`. Each node in a simulation corresponds to an instance of the class `Node` and its corresponding layers. As mentioned above, the user has the option to use or not a protocol in any of the layers, which is then automatically filled with a debugging class if the user does not specify one. However, at least a class extending the `ApplicationLayer` has to be provided, in order to let the user to define the semantics of the simulated application. The `SimulationManager` has a collection of all nodes needed in a simulation. The interface `EnergyManager` provides the template to model energy managers for the nodes. The class Link represents the bidirectional link between two nodes and is defined by the identifiers of two nodes.

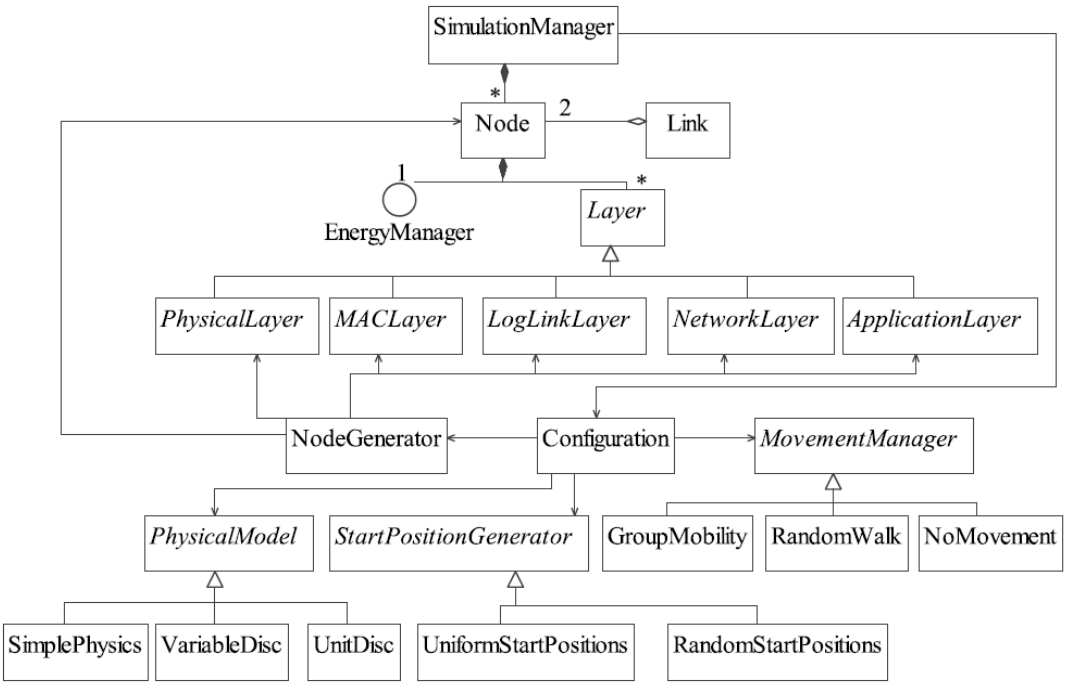


Figure 6.3: Class Diagram for the nodes' structure.

During the initialization phase of a simulation, all node instances are generated by the `NodeGenerator`. The `Configuration` class handles the simulation configuration based on information in the XML-based configuration file. The `MovementManager` is an abstract class for all classes that model movement patterns.

In Figure 6.3 there are three examples, the `GroupMobility`, used by simulations in which there are nodes with different movement patterns, the `RandomWalk`, which model random movement pattern, and the class `NoMovement`, used for nodes that are static. The `StartPositionGenerator` is an abstract class that provides a template for classes that models the distribution of nodes in the simulated environment. An example of a concrete class derived from the `StartPositionGenerator` is `RandomStartPositions`, which randomly distribute the nodes in the environment. The class `PhysicalModel` is an abstract class that provides a template needed for the implementation of propagation models for the wireless communication. The class `UnitDisc` for instance implements the omnidirectional radio wave propagation model.

6.2 Implementation of the proposed solutions using GrubiX

GrubiX is a general purpose ad hoc network simulator and does not have specific support for agent-oriented programs or their development yet. Therefore, the mobile software agents used in the proposed solutions presented in this thesis work (missionAgents, beeAgent and alarmAgents) are implemented as ordinary Java objects. These objects are transmitted among the nodes via a special type of communication packet that extends the class `ApplicationPacket` provided by the simulator framework. The nodeAgent has its logic implemented in a class that extends the `ApplicationLayer` for each type of sensor node. This extension of the `ApplicationLayer` is also responsible to run the logic encapsulated in an object that represents a mobile agent when it receives an `AgentPacket`. The ***inform*** and ***request*** agents' actions are implemented with ordinary get and set object methods. All simulations use the class `UnitDisc` for the physical model, for which the parameters `reachableDistance` and `interferenceDistance` are setup with the same values. These parameters define the communication range of the nodes. For the physical and MAC layers, the classes that define the IEEE 802.11bg standard are used. Class diagrams representing the structural models of the implemented simulations are presented in the following. These diagrams are simplified versions of the complete models, showing only the main classes and relations among them.

6.2.1 Sensing Mission Dissemination and Allocation in Static WSN

Figure 6.4 presents the class diagram for the main classes that implement the simulation reported in Section 3.4. Names written in **boldface** font represent classes provided by the simulator framework (e.g. **WakeUpCall**), while names in ***italics*** font represent abstract classes (e.g. ***MobileAgent***). The diagram presents a simplified model, in which only the main classes and their main attributes and methods, as well as relation between classes, are displayed in order to not overload the figure.

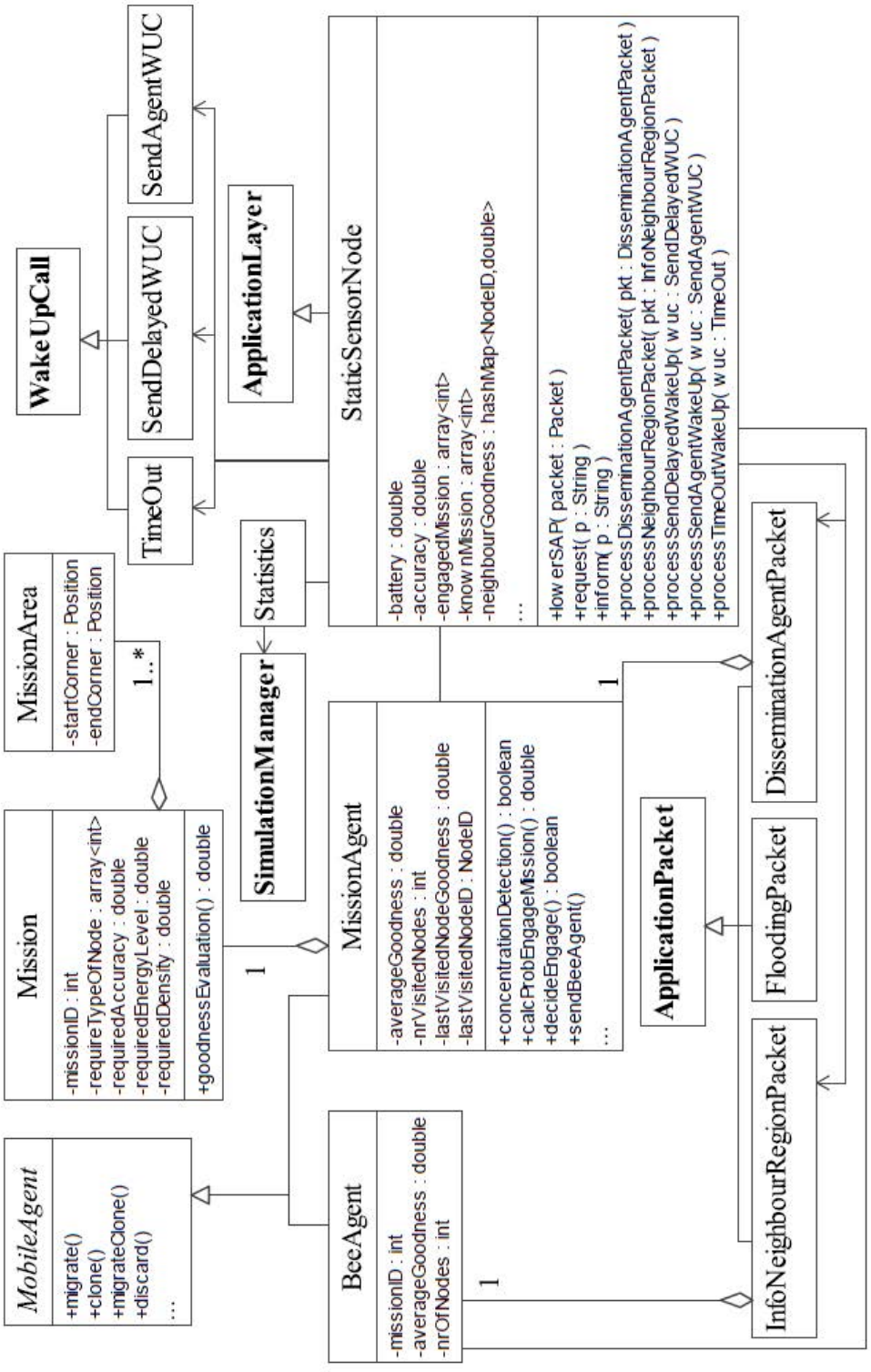


Figure 6.4: Class Diagram of the software developed for the simulation of the Mission Dissemination in Static WSN.

The class `Mission` encodes the parameters and the function that composes a sensing mission, as described in Section 3.4. These parameters and the function represent a subset of the possible elements that can be used to specify a sensing mission, as described in Section 3.1.1. Each missionAgent is responsible for one mission. The class `MissionAgent` inherits from the abstract class `MobileAgent`, and presents attributes that store information that they collect from the sensor nodes while performing the mission dissemination, as explained in Section 3.2. Its methods perform functionalities related to: identification of homogeneous regions `concentrationDetection()`, sending of beeAgents `sendBeeAgent()`, calculation of if to engage `calcProbEngageMission()` and decisions about the engagement `decideEngage()`.

The class `BeeAgent` also extends the `MobileAgent` class, and presents attributes that carry information about the region from where it is being sent.

Mobile agents are sent via specific application packets. The `DisseminationAgentPacket` is used by the missionAgent to disseminate the mission, while the `InfoNeighbourRegionPacket` is used by the beeAgent. This possible semantic difference of `ApplicationPackets` is a feature provided by the simulator, which allows a better organization of the code by separating the processing of each type of packet in different methods, which usually have the following prototype: `process<NameTypePacket>(pkt: <NameTypePacket>)`. This pattern is also valid for other events, such as wakeup calls.

The class `StaticSensorNode` that extends the `ApplicationLayer`, implements the functionality of the nodeAgent. Besides implementing the nodeAgent functionality, it also executes the functionalities of the missionAgent and the beeAgent, when the node receives a `DisseminationAgentPacket` or an `InfoNeighbourRegionPacket` respectively. This is done by the corresponding methods:

`processDisseminationAgentPacket(pkt: DisseminationAgentPacket)`

and

`processInfoNeighbourRegionPacket(pkt: InfoNeighbourRegionPacket)`.

The timers used by the missionAgent before its migrations, during the mission allocation procedure and before sending a beeAgent, are implemented by means of classes that extend the `WakeUpCall` class. The class `TimeOut` represents the timer used in the mission allocation procedure, while the class SendDelayedWUC is used upon the migration of a missionAgent. The class `SendAgentWUC` is used upon the sending of a beeAgent. When the timer related to each wakeup call expires, they trigger their respective process method in the class `StaticSensorNode`.

A class `Statistics` implements the collection of the metrics reported as results by accessing the `SimulationManager`, and to get global information about the simulation to calculate the optimal reference results. The class `FloodingPacket` and the corresponding process method in the `StaticSensorNode` are used to implement the flooding-based reference.

6.2.2 Sensing Mission Dissemination in Mobile WSN

Figure 6.5 presents the class diagram for the main classes that implement the simulation reported in Section 4.3. The same notation mentioned above for the boldface and italics fonts is used in this figure. Again the figure presents only the main elements of the model.

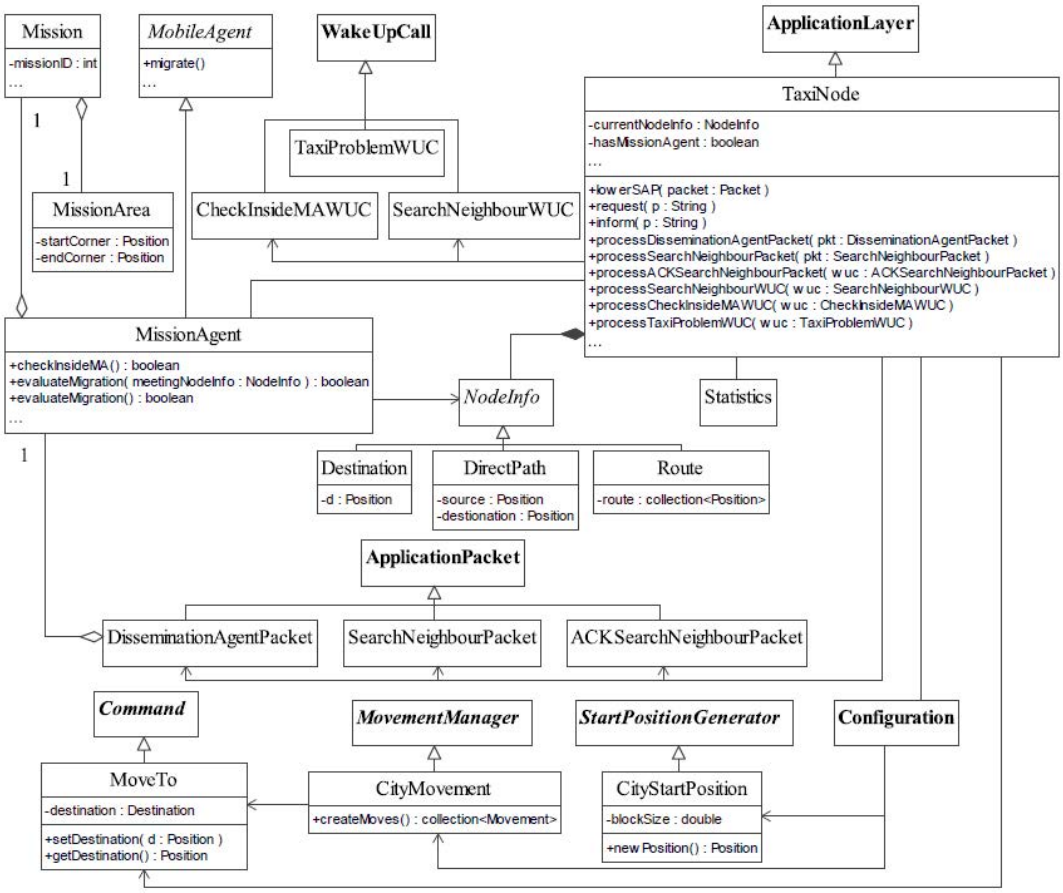


Figure 6.5: Class Diagram of the application specific software developed for simulation of the Mission Dissemination in Mobile WSN.

As in the performed simulations reported in Section 4.3 the semantic of the mission is not relevant, as the goal is only to assess the proposed mechanism to move and keep it inside the mission area. Hence, the class Mission just presents as attributes the information about its `missionID` and the information about its respective mission area.

The class `MissionAgent` presents a method `evaluateMigrate (meetingNodeInfo : NodeInfo)`, which provides the functionality described in Section 4.2, according to the three different proposed intelligent reasoning levels and a method to check if it is inside or outside the mission area (`checkMA()`). The level that will be used depends on the type of the information that is provided by the nodeAgent,

which has its functionality implemented in the `TaxiNode` class that extends the `ApplicationLayer` class. The class `NodeInfo` is an abstract class that is extended by three other classes called `Destination`, `DirectPath` and `Route`, each one corresponding to the data to be used by each of the different intelligent reasoning levels. For the random-based reference solution, the method `evaluateMigrate` is overloaded requiring no parameter and providing a randomly chosen Boolean value as result.

The class `TaxiNode`, besides implementing the functionality of the nodeAgent, executes the missionAgent. After receiving a `SearchNeighbourPacket` it replies with a packet called `ACKSeachNeighbourPacket` in case it does not carry a missionAgent and the `NodeInfo` requested by the sender node. If the missionAgent decides to migrate to the meeting node, this node receives a `DisseminationAgentPacket` and runs the appropriate method to process it, i.e. `processDisseminationAgentPacket(pkt: DisseminationAgentPacket)`. This method schedules a `CheckInsideMAWUC`, which will execute the method `checkInsideMA()` of the class `MissionAgent` in its respective process method `processCheckInsideMAWUC(wuc: CheckInsideMAWUC)`. If this method return false, a new wakeup call is scheduled, the `SearchNeighbourWUC`. This one will keep sending a `SearchNeighbourPacket` when its respective timer expires.

The model of the environment and the movement pattern described in Section 4.3.1 are implemented by the classes `CityStartPositions`, `CityMovement` and `MoveTo`. The first of these classes extends the abstract class `StartPositionGenerator` provided by the simulator framework. This class takes the size of a block and divides the area accordingly so that an environment like the example depicted in Figure 4.4 is achieved. The result of the method `newPosition()` called by the `Configuration` class is an object of the class `Position` that provides $\boldsymbol{x}$ and $\boldsymbol{y}$ coordinates in the lines defined by the division of the area according to the size of the blocks. The `CityMovement` class extends the abstract class `MovementManager` and implements the method `createMoves` which returns a collection of objects `Movements`, which contains the next movement steps of each node in the simulation. These movement steps provide the next positions which the nodes will move to until they reach a destination position. As explained in Section 4.3.1, this destination is randomly chosen for the case of taxis without passengers, while it is provided by the object of the class `MoveTo`, which extends the abstract class `Command`, for taxis with passengers. The class `Command` provides a template to construct classes that allow other objects besides the `Configuration` or the `SimulatorManager`, to pass values to an object of a class that extends `MovementManager`. In the implementation of this simulation, the value passed is the destination, which is a position randomly chosen when the taxi is taken by a passenger. The decision about if a node is taken by a passenger is also randomly taken. It is processed by the method `processTaxiProblemWUC(wuc : TaxiProblemWUC)`, which is triggered by the `TaxiProblemWUC` wakeup call.

6.2.3 Alarm Handling in Cooperative Static and Mobile WSN

The class diagram for the implemented simulation used in the experiments reported in Section 5.5 is presented in Figure 6.6. The figure follows the same notation of the ones presented in the above two subsections. Despite the importance of the relations between classes, the graphical representation of some of them is omitted in the figure in order to not overload it and compromise its readability.

The model presents three classes extending the `ApplicationLayer`. The first one is the `StaticSensorNode` class, which implements the functionalities of the nodeAgent for the static sensor nodes. The second class is called `UAV` and implements the functionalities of the nodeAgent for the mobile sensor nodes (carried by UAVs). The third class models the behaviour of threats in the simulation. The threats are simulated in a way similar to nodes in the network, but they do not participate in any communication.

As explained in Section 5.3, the UAVs send beacon messages with pheromones that are stored by the static sensor nodes on the ground. This behaviour is implemented by the class `PheromonePacket` which is sent by the UAV to tell about its current position as an attribute contained in an object of the class `Pheromone`. This class defines that the pheromone carries the `NodeID` of the `UAV`, the pheromone level (which is maximum when the UAV is sending it) and the flavour, which is the same for all UAVs in the case of the simulations presented until Section 5.5.5 inclusive. For the simulations presented in Sections 5.5.6 and 5.5.7, the attribute flavour has different values according to the attribute type of the class `Sensor`. The class `Pheromone` has methods to manipulate the pheromone level, which are used to: reset the pheromone level when a static sensor node receives the pheromone of a UAV that it had already received, method `resetLevel()`; to diminish its level due to the time elapsed since a static sensor node received the pheromone, method `evaporate()` used by the `processPheromoneDecreaseWUC(wuc:PheromoneDecreaseWUC)` activated by the respective wakeup call; and to diminish the pheromone level when propagating a trail, methods `propagateTrails()` and `processTrailPropagationPacket(pkt:TrailPropagationPacket)`, as explained in Section 5.3.2. The variations in the trail propagation process presented in Section 5.3.2 provide different implementation of these methods.

The static sensor nodes periodically measure and compare the environment to some ground truth (e.g., defined by thresholds) in order to detect threats. This periodic behaviour is implemented by a wakeup call called `MeasureWUC`, and is then processed by the method called `processMeasureEnvironmentWUC(wuc:MeasureWUC)` in the class `StaticSensorNode`. The threat detection is implemented by searching among all the neighbour nodes within the sensing range of the node that is performing the measurement. This aims to identify if there is any of these neighbours has a value that characterizes a threat in the attribute `typeOfNode`. The attribute `typeOfNode` is defined in the class Node, which is the basic class for all the nodes in the simulation. To perform this assessment, an instance of the class Transponder in the `StaticSensorNode` accesses the `SimulationManager` to get information for all

these nodes within the sensing range. If a node in the sensing range is of the type "Threat", the detection is confirmed.

When a static sensor node detects a threat, it reads the threat type from the node that represents the threat and takes the value for the weather conditions in its position (attribute `weatherCondition`) to instantiate an object of the class `Alarm`. The value for the weather condition is a Boolean value randomly selected in the simulation initialization for each node of the type `RegularNode` (the type attributed to the static sensor nodes), in which true means bad weather (changes the UAVs' sensors applicability) and false means good weather (no change in the UAVs' sensors applicability). An object of the class `Alarm` is instantiated with these data together with the `NodeID` of the alarm issuer node and its position, which is encapsulated in an object of the class `AlarmAgent`, and then sent to the network via an object of the class `AlarmPacket`. This object of the class `AlarmPacket` is received by the neighbour nodes and processed by the method `processAlarmPacket(pkt:AlarmPacket)`. By using the methods inherited from the class `MobileAgent` and its own methods `comparePheromoneLevel()` and `updatePheromoneLevel()`, the object of class `AlarmAgent` then performs the trail-search and trail-follow mechanisms as described in Section 5.3.

An `AlarmPacket` received by a UAV, is processed by the method `processAlarmPacket(pkt:AlarmPacket)`, which will store the alarm delivered by the alarmAgent in a list of alarms (`pendingAlarms`) and check the status of the UAV. If the status is idle it will call the method `moveUAVToThreat()`, otherwise it will follow the previous alarm handling that it was engaged to accomplish. The `AlarmAgent` is then discarded as it has accomplished its mission. The UAV send then a `UAVAlarmACKPacket` to the static sensor nodes, so that they will stop the forwarding of the alarmAgent for that alarm, which is discarded by all the static sensor nodes. This process to discard the alarmAgent of a alarm received by a UAV is performed by the `processUAVAlarmACKPacket(pkt:UAVAlarmACKPacket)` method.

The `moveUAVToThreat()` method takes the position informed in the alarm and calls the method `setDestination(d:Position)` passing this informed position as parameter. As a result, the movement pattern defined in the class `ComposableRandomWalkWithCollisionAvoidande` will change from a conventional random walk as described in Section 5.4 to a straight line linking the current nodes position and the position where the threat was identified. Once the UAV arrives to the position in which the threat was detected, the threat is considered handled and the UAV will use the method `handleAlarm()` to get the next alarm from its `pendingAlarms` list.

The class `ComposableRandomWalkWithCollisionAvoidance` also uses an object of the class `Transponder` to avoid that two nodes of the type UAV collide, i.e. occupies the same position at the same time instant.

The UAVs use a wakeup call to send pheromones to ground sensor nodes, the `LayPheromoneWUC`. As presented in Section 5.3.2, the UAVs may also send pheromones of other UAVs together with their own in a `PheromonePacket`. This is

the case when a UAV previously received a `TrailPropagationPacket`, which is processed by a corresponding process method (`processTrailPropagationPacket(pkt:TrailPropagationPacket)`). This method stores the received pheromone in a list containing other UAVs' pheromones (attribute `friendlyPheromones`). When the timer related to the `LayPheromoneWUC` expires and its process method is called, pheromones of other UAVs stored in the `friendlyPheromones` are updated by the method `updateFriendlyPheromones()` before the UAV then sends the `PheromonePacket`. The method `updateFriendlyPheromones()` uses the method `decrease()` from the class `Pheromone` to adjust the level of the pheromones of other UAVs before sending them.

An object of the class `Statistics` collects all results in terms of the number of packets sent, and the utility of UAVs to handle a given threat that are reported and presented in Section 5.5. The optimal reference solutions uses class called `OptimalSourceRouting` that implements the short path routing to the closest UAV that fulfils the requirements of the different optimum criteria (optimum-C: the closest UAV, optimum-T: the closest UAV idle or with smaller `pendingAlarms` list, and optimum-U: the closest UAV with the maximum utility value). To implement shortest path routing this class accesses information about all nodes in the `SimulationManager`, and select those that provide the shortest path towards the selected UAV.

The flooding based solution is implemented by the `FloodingAlarmPacket`, which is addressed and is forwarded to all nodes in the network, as explained in Section 5.5. The mentioned conditions to stop the flooding are implemented by assessing the `SimulationManager` from the method that process the `FloodingAlarmPacket` (`processFloodingAlarmPacket(pkt:FloodingAlarmPacket)`) in the static sensor nodes.

6.3 Summary

This chapter presented the details about the tool used to implement the simulations performed in this thesis, as well as how these simulations were implemented using the resources provided by this tool. The mapping of important concepts presented in Chapters 3, 4 and 5 to elements of the implemented simulations were highlighted. The reuse of these elements by the different simulations could also be noticed, such as the classes `Mission` and *`MobileAgent`*. The way the agents' behaviours and interactions were implemented was explained, mentioning the corresponding methods responsible by them and their semantics according to the framework, as well as how trigger conditions were implemented by means of wakeup calls. These details clarify how those high level concepts presented in the three previous chapters were implemented in the GrubiX simulator.

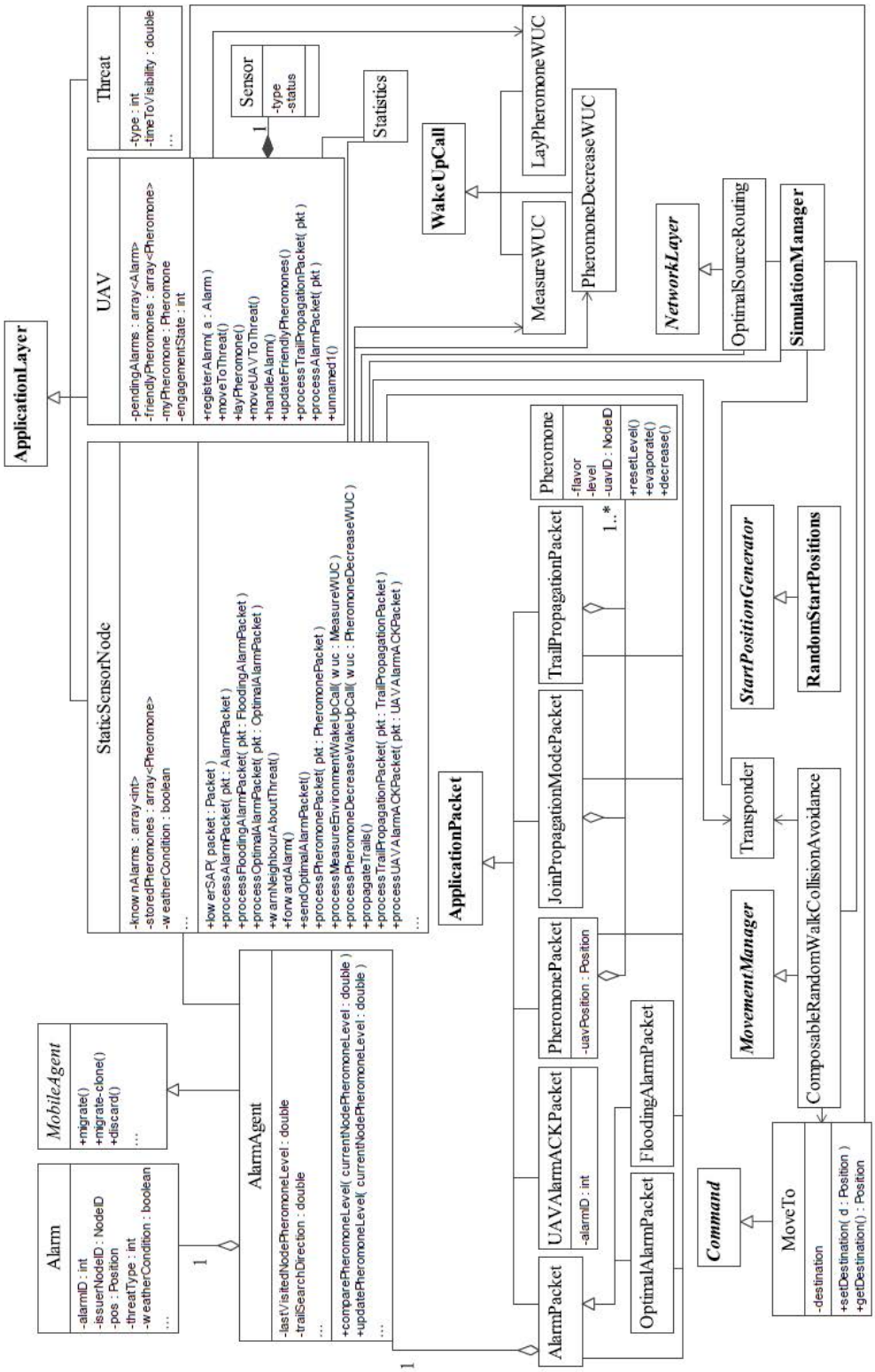


Figure 6.6: Class Diagram of the software developed for the simulation of the proposed biologically-inspired approach for cooperation among static and mobile sensor nodes.

7 DEMONSTRATOR

As mentioned in Section 1.7, simulation is the main method used to perform experiments to test the solutions proposed in this thesis work. However, in order to assess the real world feasibility of our proposal using software agents in the presented context, a small scale demonstrator has also been implemented for a selected mechanism proposed in this thesis. The selected mechanism deployed in the demonstrator is the trail-follow presented in Section 5.3.

There are two main goals in the assessment performed in this demonstrator. The first is the assessment of the feasibility of the proposed approach considering a full featured agent-oriented software platform. This type of assessment is not possible to be done using GrubiX, as it does not provide yet the entire support needed for agent-oriented programming, as already mentioned. The second goal is to deploy the system in a commercial of the shelf (COTS) sensor node platform, so that it is possible to assess the feasibility in terms of a real world deployment.

It is important to remark that the demonstrator has no ambition to prove or assess any financial suitability or budget estimation, but only assesses the technical aspects of the feasibility of the proposed approach in a small scale physical system, using COTS devices.

7.1 Demonstrator Design

The demonstrator is a small indoor setup consisting of a network of static sensor nodes distributed on the ground and a one mobile sensor (UAV) flying over the ground sensor nodes. The operation of the network is to be exactly the same with respect to the trail-follow mechanism, i.e. the mobile sensor node sends beacons to the static sensors on the ground, which are handled by the static sensor nodes as indication of the mobile sensor location, by using the pheromone-based mechanism presented in Section 5.3.

Figure 7.1 illustrates the static sensor nodes distribution and the entrance of the mobile one in the communication range of one of the static sensor nodes. As can be observed in the figure, twelve static sensor nodes are displaced on the ground forming a grid-like distribution, in which they are placed 75 cm apart from each other. The number in the centre of the circle that represents a sensor node is its identifier. Their communication range is configured to reach neighbour nodes on the south, north, east and west, but they are not able to communicate with the neighbours located in the diagonals. For example, sensor node number 6 is only able to communicate with sensors

7 (south), 5 (north), 10 (east) and 2 (west). The communication is restricted in this way due to the small number of nodes in the demonstrator; otherwise each sensor node would be able to communicate with almost every other node in the network, which would not allow a realistic test.

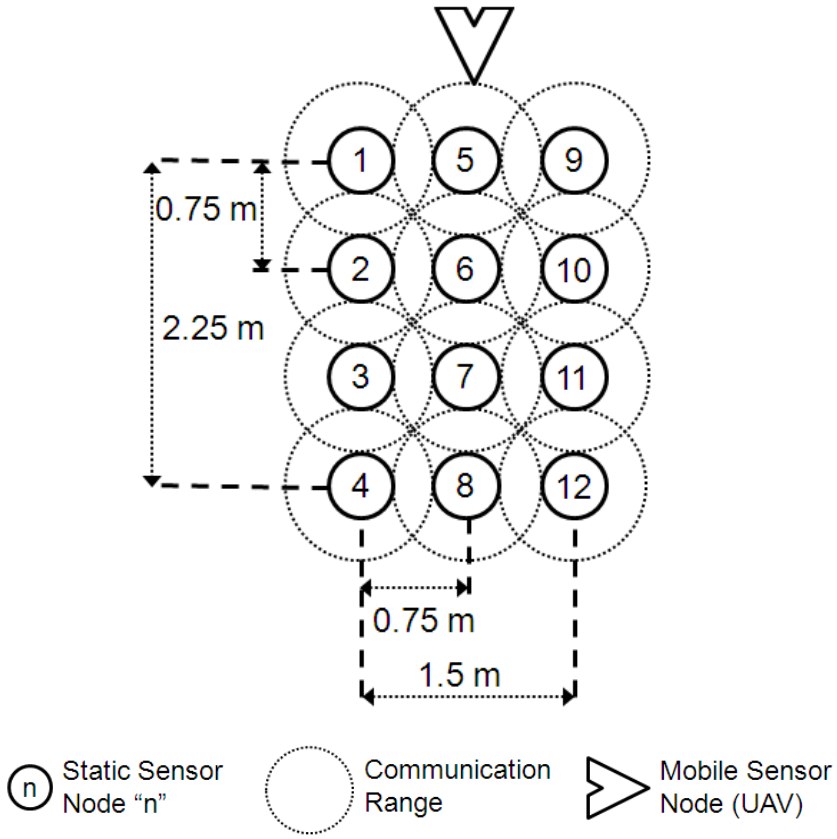


Figure 7.1: Top view illustration of the demonstration setup.

To keep it simple, the mobile sensor node is the same device that is used of the sensors on the ground, which is carried by a remote controlled mobile platform. Details about the sensor node device, the mobile platform and the middleware used to implement the trail-follow mechanism are presented in the following section.

7.2 Hardware and Software Components

7.2.1 Hardware

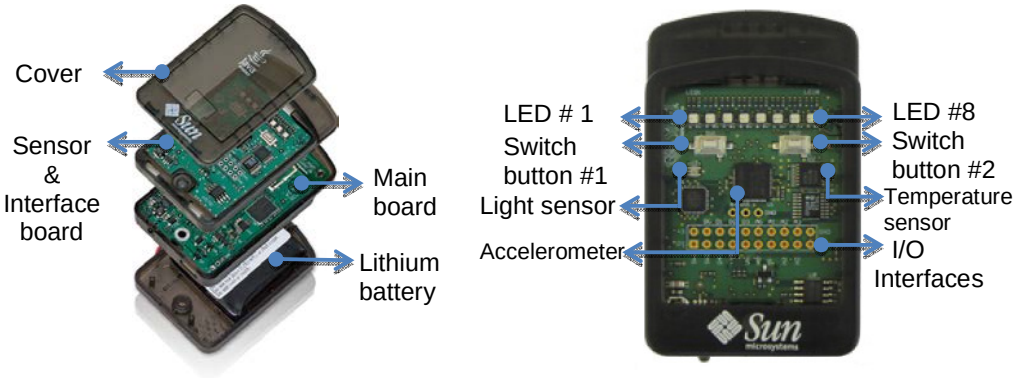
The hardware used to build the demonstrator is composed of SunSPOTs (SUN, 2010) as the sensor nodes and a remote controlled quadcopter as mobile platform for the mobile sensor node. The SunSPOTs are selected as the COTS sensor nodes because they are based on the Java technology that provides an easy process to develop and deploy new applications. The quadcopter chosen is a very stable platform, which

enables its usage both indoors and outdoors, thus providing a great flexibility to perform tests in different scales.

7.2.1.1 Sensor Node

SunSPOT (Sun Small Programmable Object Technology) (SUN, 2010) is an embedded platform developed by Sun Microsystems Laboratories, which was overtaken by Oracle in 2009, to program Java applications for Wireless Sensor Networks. It is a small programmable wireless device that is able to communicate and interface with other system parts on a network level using the IEEE 802.15.4 MAC and PHY layers over a Wireless Personal Area Network (WPAN).

The SunSPOT is based on a 32-bit 180 MHz ARM920T core that includes 4 MBytes of flash memory and 512 KBytes of RAM memory. It has a built-in Lithium battery with 720 mAh capacity. Figure 7.2a shows a SunSPOT platform and all of its components divided in different layers, i.e. from bottom to top, the energy source, the processing board, the sensor and interface board and the cover layer. It has an USB interface that is used to charge its battery and communicate with a software management tool installed in a desktop computer. It is equipped with a temperature sensor, a 3 axes accelerometer and a light intensity sensor. For additional interfacing to humans and devices, it has 8 3-colours LEDs, 2 switch buttons and 4 general-purpose I/O pins, and 4 high current output pins. The sensor and interface board is shown in Figure 7.2b.



(a) (b)

Figure 7.2: SunSPOT sensor node platform: a) component layers; b) Sensor & interface board (figures adapted from (SUN, 2010)).

The SunSPOT platform runs Java natively, i.e. without operating system, and it has a J2ME compliant Java VM called Squawk running on the device. However, it does not support the full Java library and only has a limited number of classes and data structures. It can run many mutually isolated applications in a single VM. The memory management is supported by a garbage collection mechanism that is able to efficiently trace references optimizing the use of memory space. The devices such as sensors and interfaces are represented by Java objects whose features are accessed through their methods.

The SunSPOT kit come with a tool for its management called Solarium. Solarium provides means to update software on the SunSPOT nodes, as well as to monitor their status, in terms of memory usage and battery level. This tool connects with the SunSPOTs via USB cable or radio connection provided by the SunSPOT base station connected to the PC on which Solarium is running. It provides also an environment to emulate SunSPOT nodes, creating virtual nodes that act like real SunSPOT nodes. It is possible to deploy, run and debug programs in the virtual nodes in the same way as it is done in the real ones. Besides, it is possible to create a heterogeneous network composed of real and virtual SunSPOT nodes. Figure 7.3 shows this setup possibility.

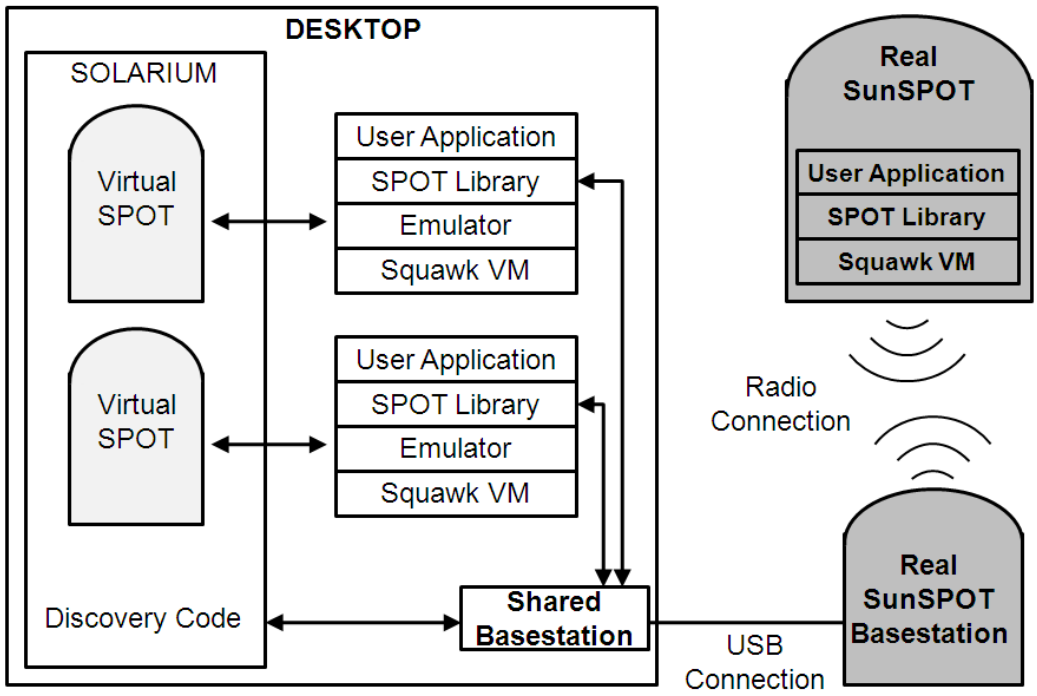


Figure 7.3: Solarium environment block structure: heterogeneous network with both real and virtual SunSPOT nodes (figure adapted from (SUN, 2010)).

7.2.1.2 Mobile Platform

The mobile platform chosen for the demonstrator is a small quadcopter, a flying platform that has four propellers, equipped with a flight control and navigation assisted by GPS. The quadcopter used in the demonstrator has its electronics developed by the German company Mikrokopter (MIKROKOPTER, 2011), and adapted by the Brazilian company Skydrones (SKY, 2011). Figure 7.4a presents an illustrative picture of this platform ready to fly, while Figure 7.4b presents its internal electronic devices.

(a) (b)

Figure 7.4: Quadcopter platform: a) quadcopter and remote control; b) internal electronic devices.

The platform weights 850 g and has its dimensions defined by the size of the four supporting bars in which the propellers are fixed. Each bar has 20 cm length. It is equipped with five sensors: a gyroscope for each axe, a compass, a 3 axes accelerometer, a GPS and a pressure altitude sensor. The flight control system run on an Atmel ATMEGA644 - 20MHz microcontroller (Flight Control Board) while the navigation control runs on an ARM-9 processor (Navigation Control Board). It uses a LiPO 5000 mAh battery that is the energy source for the propellers and the electronic devices, providing endurance for between 15 and 20 minutes of flight. A 6 channel radio system is used for remote control.

The choice for this platform is based on the fact that it is very stable and easy to operate. Its stability is due to the manipulation of data from the 3 axes accelerometers and gyroscope by the flight control unit. These features allow its usage both indoors and outdoors. When used indoors, it must be remote controlled, as it is not able to use the automatic navigation control assisted by GPS.

To use the quadcopter as the mobile sensor in the demonstrator, a SunSPOT sensor node is mounted on top of the cover that protects its electronic devices, as shown in Figure 7.5.

Figure 7.5: SunSPOT sensor node mounted on top of the quadcopter platform.

7.2.2 Software

To implement the pheromone-based alarm delivery on the SunSPOT sensor nodes, a framework called AFME (Agent Factory Micro Edition) (MULDOON, 2008) is used. It is based on Java technology and it is an agent platform characterised by a minimised footprint, which provides support to agent migration and for deployment in the SunSPOT platform, thus providing the features that are necessary for the development of the designed demonstrator.

AFME is a low scale agent framework, which was developed to enable the creation of agent systems for mobile devices and resource constrained devices. It is designed to handle the Constrained Limited Device Configuration (CLDC)/Mobile Information Device Profile (MIDP) subset of the Java Micro Edition (J2ME) specification. AFME is based on Agent Factory (O'HARE, 1996), a large framework used for the deployment of multi-agent systems. The framework is complaint with FIPA specification enabling its interoperability with other FIPA-compliant environments. AFME uses a rule-based concept similar to expert systems (GIARRATANO; RILEY, 1989) to represent the agents' behaviours and maintain a reduced set of meta-information about itself and its surrounding environment. This information represents the agents' beliefs. Rule-based operations over the agents' belief set determine the agents' commitments, which finally drive what actions that the agents must do.

The AFME development process requires that several components must be developed so that an agent platform can be built. These components are compiled together with an agent platform script, generating Java files as output. These files are then used to build the MIDlets for SunSPOTs or similar devices. The main components that have to be developed are: perceptors, actuators, modules and services.

Perceptors are Java objects that generate beliefs about the agent's state and its environment. There are two other ways that can generate beliefs, one is being inserted in

the agent platform script by the developer (this type is called initial belief), or beliefs can be added by actuators.

Actuators enable agents to affect their environment by means of performing actions, i.e. they provide the imperative functionality for primitive or atomic actions. The action or set of actions that an agent will perform is decided by the manipulation of its beliefs by its internal declarative rule set.

Perceptors and actuators are decoupled, i.e. they do not contain direct object references to each other. However, it is very probable that they have to exchange information, for instance when an actuator is able to update a state of an object that a perceptor has to perceive. To address this issue, a component called Module is used. Modules allow not only information sharing between a perceptor and an actuator, but among perceptors and actuators themselves too.

However, modules are only visible within the scope of a single agent, and in a multi-agent system, it is highly probable that agents need to exchange information. This issue is addressed in AFME by the use of services. Services enable multiple agents to access shared objects by using their perceptors and actuators. Figure 7.6 illustrates the AFME architecture, in which it is possible to observe how agents access services. Agents are scheduled when to execute by the scheduler. When executing, they first update their beliefs by using their perceptors and then fire a reasoning process based on these updated beliefs. Depending on their desires, they adopt commitments that may fire several actuators. During this entire process, they may need to access information from other agents (perceive) or share information with other agents (act). The access to the platform services allows this communication among the agents to be performed.

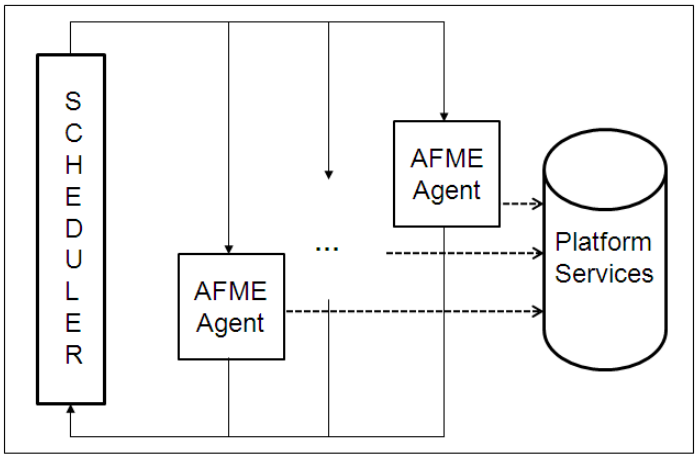


Figure 7.6: AFME Architecture (MULDOON, 2008).

An important service provided in AFME is the migration service. This service allows the migration of an agent from one AFME platform to another. This service operates similarly to a message transport service (FIPA, 2002b), contacting an external server to send and receive agents.

7.3 Software Design

The UML model developed for the software deployed in the SunSPOTs present two class diagrams, one for the software deployed in the static sensor nodes and another for the software deployed in the mobile sensor node. The model includes the classes from the application itself plus the essential ones from the AFME framework. Notice that the framework has many other classes, which are not presented in the following diagrams because they were not used in the application or they are not essential for the overall understanding of the software structure. Parts of the developed class diagrams are presented in separate diagrams as follows, which were selected in order to show the relationship between the most important classes from the framework and those from the application.

The class diagram of Figure 7.7 shows the class `UAVNodeAgentPlatform` which implements the interface `Platform` from the AFME API. This class provides the basic functionalities to the agents that are hosted in a node and contains an instance of the `BasicRunnable` class, which provides the basic functionalities to execute an agent, updating its beliefs and driving the agent's control process according its rules to select the appropriate actions to be performed. The instance of this class that represents a nodeAgent in the mobile sensor node is the `UAVNodeAgent`. The interface `MigrationPlatform` is implemented by the `UAVNodeAgentPlatform` so that this node is able to receive a mobile agent to deliver an alarm. The diagram in Figure 7.7 shows also a platform service implemented by the class `AlarmDeliveryServ`, which is used to assist in the alarm delivery process, as it will be explained further.

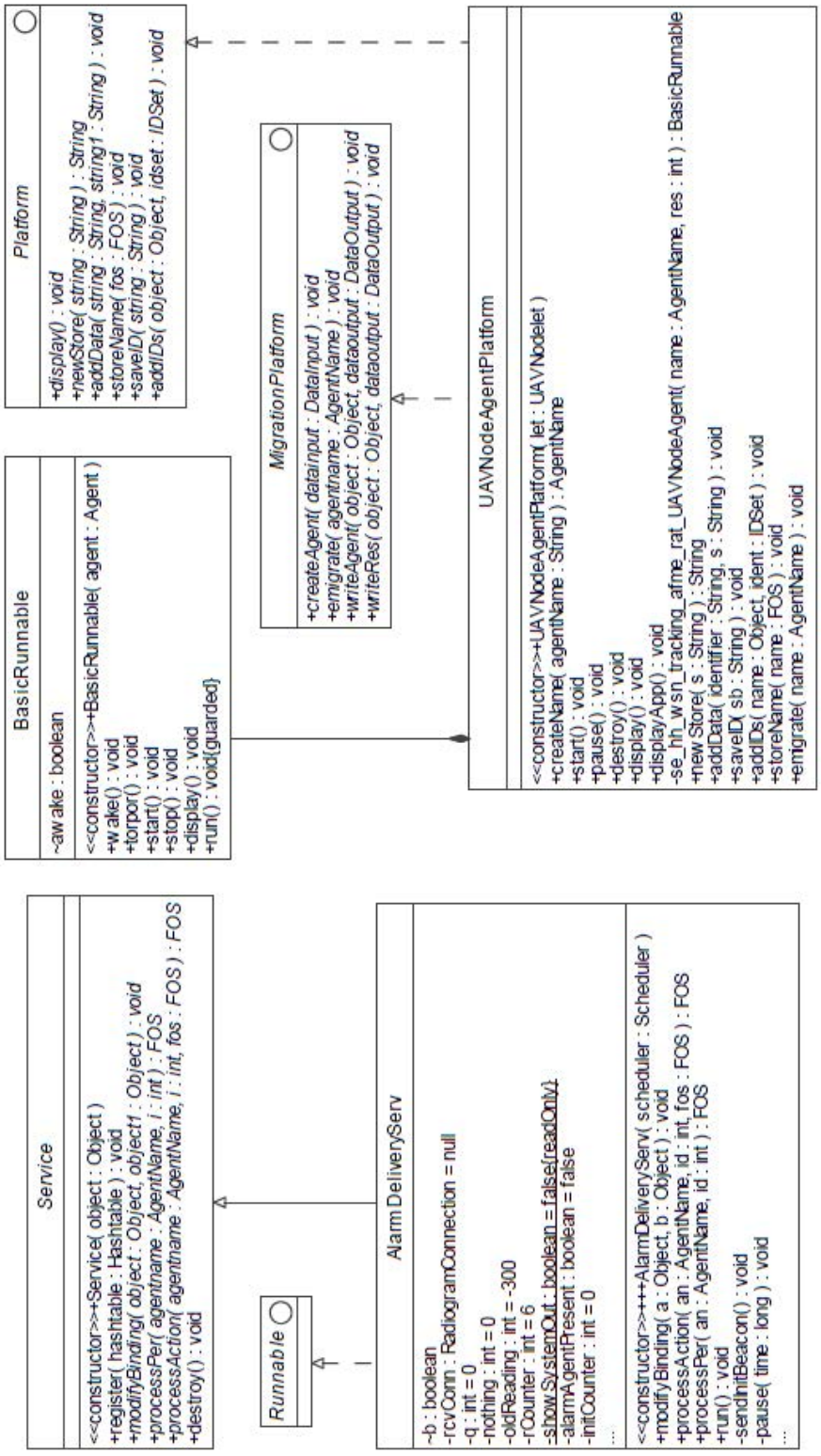


Figure 7.7: Class diagram for the platform classes supporting the mobile node.

The supporting platform for the agents in the static sensor nodes is provided by the class `SensorNodeAgentPlatform`, presented in Figure 7.8. Besides the implementation of the Platform interface, this class does also implements the `MigrationPlatform`, which provides the functionalities that are need to make agent migrations, sending and receiving mobile agents. This platform presents two instances of the class `BasicRunnable`, one is called `SensorNodeAgent`, which is the agent that control the sensor node, and the other is called `AlarmAgent`, which represents the mobile alarmAgent presented in Chapter 5. Using AFME, an object for the alarmAgent has to exist from the beginning of the system runtime, instead of being created during the system runtime. Thus, in this demonstrator one of the nodes is selected to be the one that will issue the alarm, and in this node the instance of the class `BasicRunnable` is created for the alarmAgent and started during the boot time. The other sensor nodes in the network have exactly the same `SensorNodeAgentPlatform` class, but they do not have this `AlarmAgent` instance created in boot time.

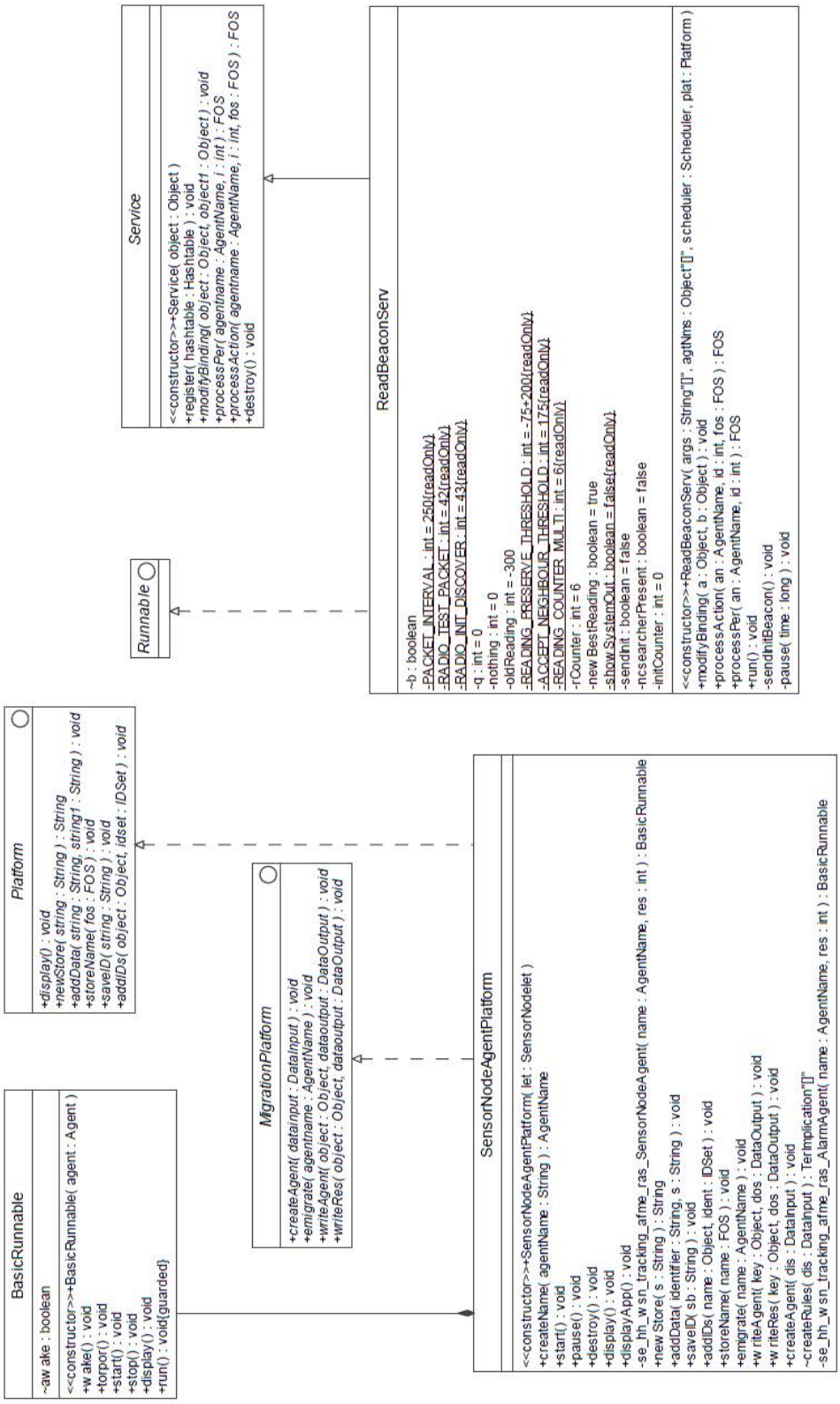


Figure 7.8: Class diagram for the classes supporting the static sensor nodes.

In the mobile sensor node, the `BeaconAct`, presented in Figure 7.9, is an actuator responsible for the action of sending beacons which will be received and stored in the static sensor nodes by the `ReadBeaconServ` service. When the mobile sensor node starts sending beacons, the `LED8RedTogAct` actuator is also activated to turn its LED number 8 on with red light.

In the static sensor nodes, the `SensorNodeAgent` perceives the pheromone information by means of the functionality of the perceptor `Check4BeaconPer` presented in Figure 7.10. This perceptor updates the agent's belief and triggers the actuator `LED8RedTogAct` (presented in Figure 7.11), which turns the LED number 8 on with red light in the static sensor node that received the beacon, to show that this node is part of a pheromone trail.

The `ReadBeaconServ` service also decreases the pheromone level according to the time evolution, as explained in Chapter 5, and manipulates the registration of an incoming `AlarmAgent` upon migration of this agent into the sensor node. If an `AlarmAgent` is registered in the `ReadBeaconServ`, the `SensorNodeAgent` uses its actuator `InformActuator` to tell the registered `AlarmAgent` about the current pheromone level.

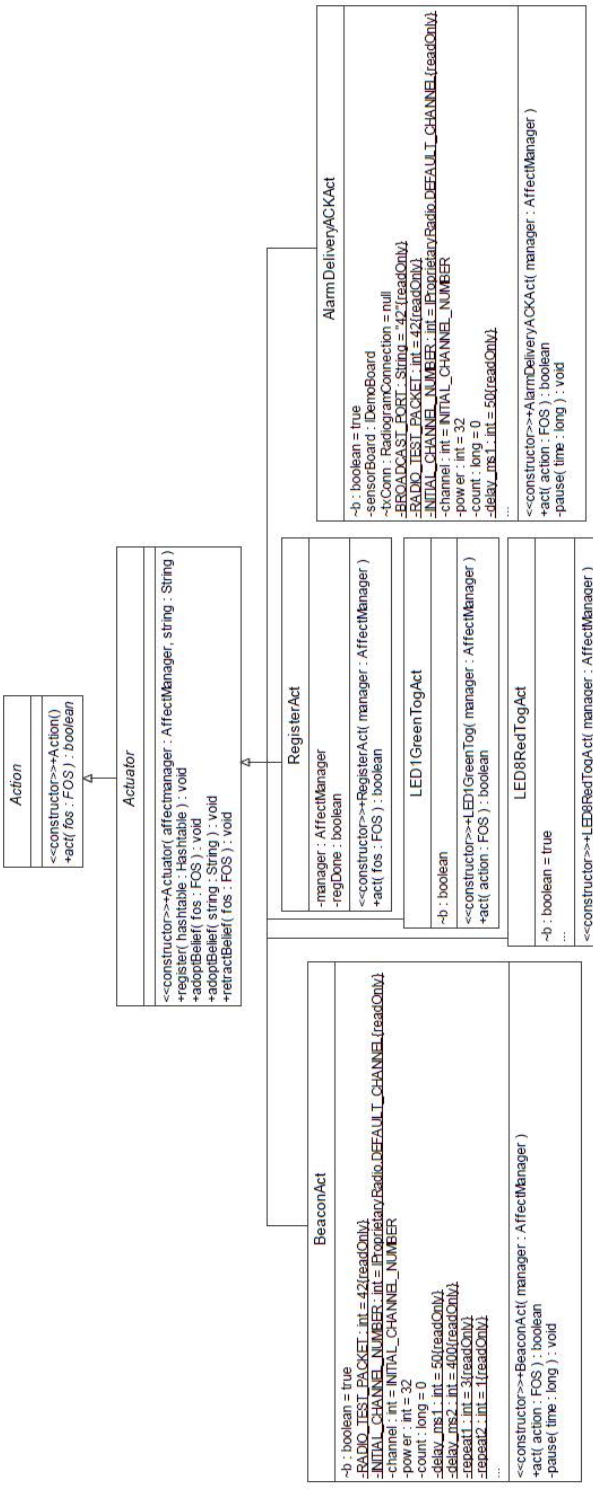


Figure 7.9: Class diagram for the classes representing the actuators of the node agent in the mobile node (`UAVNodeAgent`), and the actuator of the mobile agent (`AlarmAgent`), i.e. the `RegisterAct`.

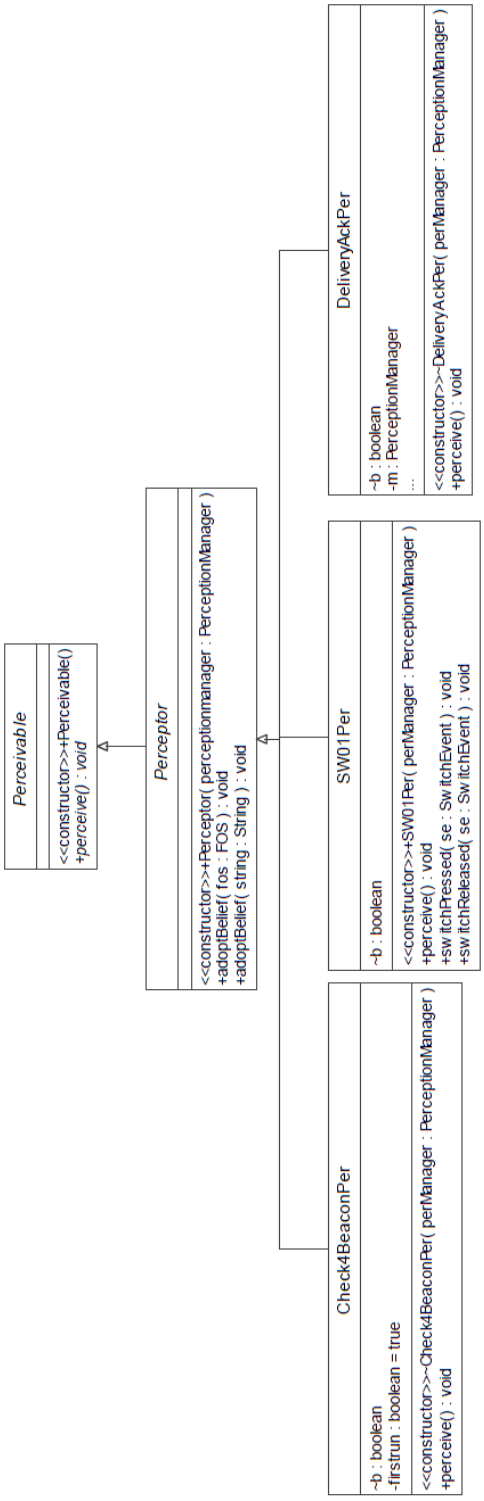


Figure 7.10: Class diagram for the classes representing the perceptors in the static sensor nodes for the `SensorNodeAgent` (`Check4BeaconPer`) and for the `AlarmAgent` (`SW01Per` and `DeliveryAckPer`).

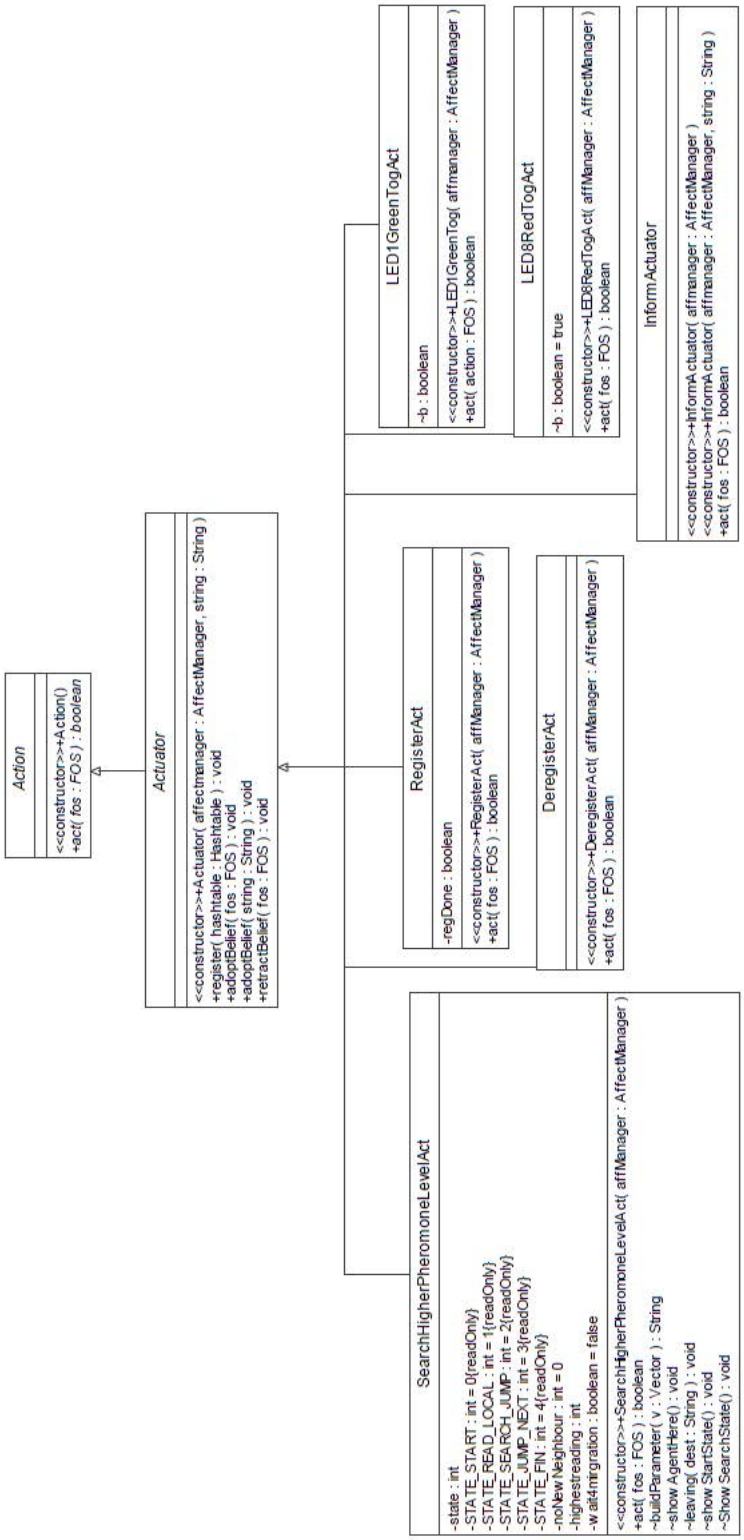


Figure 7.11: Class diagram for the classes representing the actuators in the static sensor nodes for the `SensorNodeAgent` (`InformActuator` and `LED8RedTogAct`) and for the `AlarmAgent` (the other actuators presented in figure).

In this demonstrator, an alarm is issued by pressing button number 1 of one of the static SunSPOT nodes. As mentioned above, just one node is selected to be able to issue an alarm, by having an `AlarmAgent` created and started at boot time. Thus, if the button of a sensor node that is not the one that has the `AlarmAgent` created is pressed, nothing happens. Additionally, if the button is pressed in the sensor node prepared to issue the alarm but it does not have received any beacon, i.e. has no pheromone information, nothing happens either. This is because the trail-search mechanism presented in Chapter 5 is not implemented. The demonstrator implements only the trail-follow mechanism, thus only if the sensor node received a beacon it will be able to trigger the action to send the `AlarmAgent` when having its button number 1 pressed. The event associated with the pressing of button number 1 is perceived by a perceptor of the `AlarmAgent`, the `SW01Per` (Figure 7.10). This perceptor triggers the `SearchHigherPheromoneLevelAct` actuator (Figure 7.11) of the `AlarmAgent`, which will migrate the agent to the neighbour nodes. When arriving at a neighbour node, the `AlarmAgent` will register in the `ReadBeaconServ` service (by using the `RegisterAct` actuator - Figure 7.11), and then will receive the information about the pheromone level of this node from the `SensorNodeAgent` as explained above. If the pheromone level is higher than the one of its previous hosting sensor node, the agent will activate the `SearchHigherPheromoneLevelAct` again to continue the trail-follow process. At the same time, the agent is deactivated from the previous hosting sensor node by the `DeregisterAct`.

Once an `AlarmAgent` is migrating to the neighbour nodes, and none of them have a higher pheromone level, the `SearchHigherPheromoneLevelAct` is not activated, and the agent execution is terminated. However, this situation means that this node that sent the agent is the one closer to the mobile node, i.e. is has the higher pheromone level. It also means that the mobile node will also receive the mobile agent, which will finally deliver the alarm.

When the `AlarmAgent` reaches the mobile node, it registers itself in the `AlarmDeliveryServ` (Figure 7.7), which stores the information carried by this agent about the alarm (in this implementation this information consists of the alarm issuer node identification only). The registration of this `AlarmAgent` is perceived by the `UAVNodeAgent` by means of the `AlarmDeliveryPer` (Figure 7.12), which triggers the actuator `LED1GreenTogAct` (Figure 7.9) that turns the LED number 1 on with green light in the mobile sensor node, and the actuator `AlarmDeliveryACKAct` (Figure 7.9), which sends an alarm delivery acknowledgement to the sending sensor node. The sensor node that sent the alarm delivery message perceives the acknowledgment message by the `DeliveryAckPer` perceptor (Figure 7.10), triggering the `LED1GreenTogAct` actuator (Figure 7.11) that turns the LED number 1 on with green light.

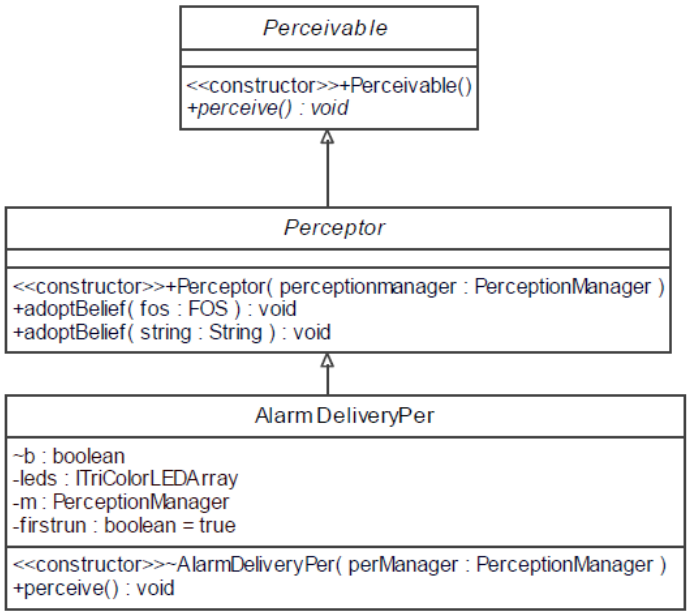


Figure 7.12: Class diagram for the class representing the perceptor in the mobile sensor node for the `UAVNodeAgent`.

7.4 Experiments and Results

Following the setup shown in Figure 7.1 and after having done the software deployment in the SunSPOTs, the performed tests consisted of the following steps:

1) Move the mobile SunSPOT carried by the quadcopter over the network in straight line from sensor node number 5 until sensor node number 7. This movement build a pheromone trail over three nodes (number 5, 6 and 7); and

2) Issue an alarm from the sensor node number 5 (the one prepared for that, as explained above), to assess the correct delivery of this alarm to the mobile SunSPOT that by this time would be positioned over the sensor node number 7.

In the deployed physical demonstrator, the sensor node number 4, see Figure 7.13, is a base station that provides access to the deployed network via Solarium management tool.

The successful operation of the proposed pheromone inspired algorithm implemented using the AFME framework and deployed in the SunSPOTs network was verified in the performed test, which is illustrated by the Figures 7.13 – 7.15. These figures present the physical demonstrator in three different stages of the test of the pheromone-based alarm delivery algorithm, described in the following.

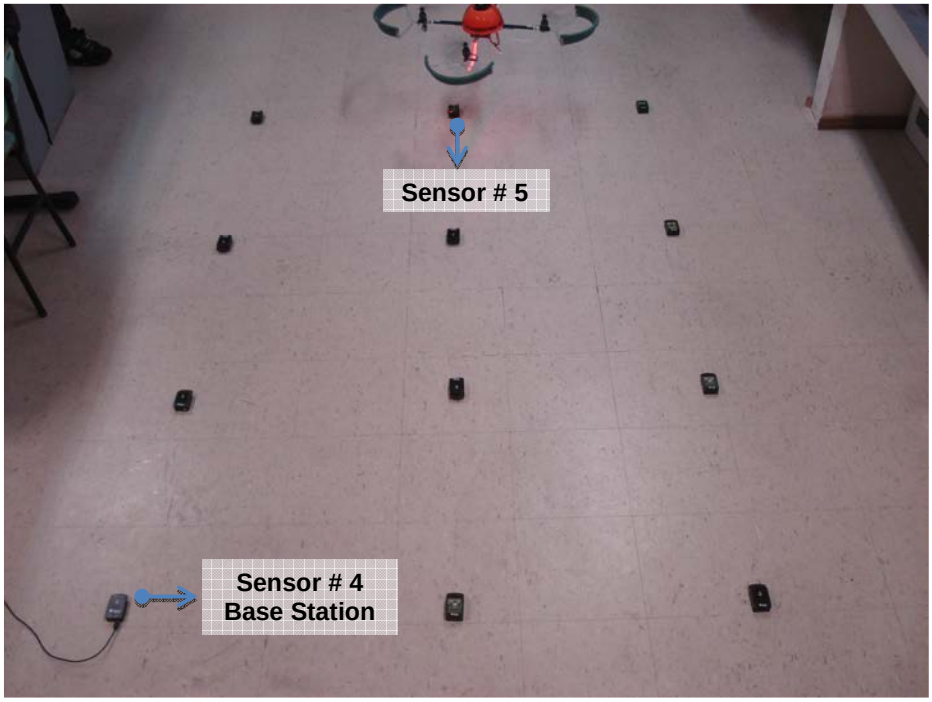


Figure 7.13: First step of the test – mobile sensor node starts sending beacons.

Figure 7.13 shows the beginning of the experiment, in which the mobile sensor node carried by the quadcopter enters in the range of the sensor network deployed on the ground. At this stage, the SunSPOT sensor node carried by the quadcopter starts to send beacons to static ones on the ground. The sensor node number 5 receives the beacon and turns its LED number 8 on with red light, indicating that it received the beacon, then storing the pheromone information and making part of the trail. As mentioned in Section 7.1, the sensor nodes have their communication range limited to avoid that they reach all their neighbours. This configuration of the communication range is also done for the mobile sensor node, so that its beacons do not reach too many of the sensor nodes, but only those that were planned to be reached to form the trail, as explained above. The configuration of the communication range is done by adjusting the radio transmission power, which is empirically performed by trying different values until getting the desired result. This need to configure the communication range can be explained by environment disturbances, such as the arrangement of the surrounding objects or the material that covers the floor.

Figure 7.14 shows the pheromone trail formed by nodes 5, 6 and 7, which can be observed by their LED number 8 on (right hand side of the SunSPOT nodes) with red light.

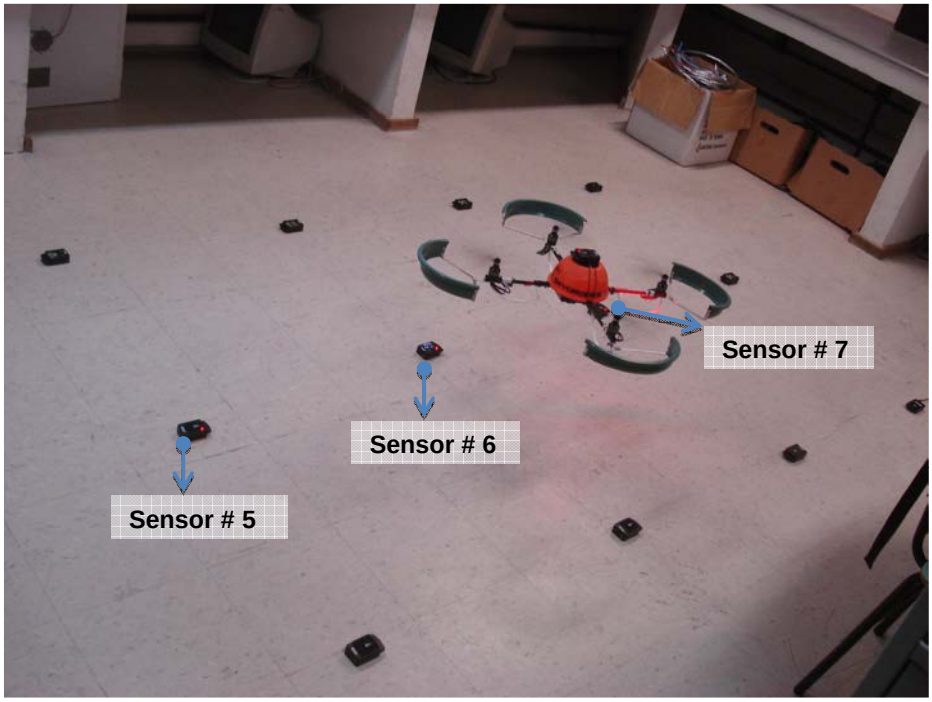


Figure 7.14: Second step of the test – pheromone trail formed by the three ground static sensor nodes.

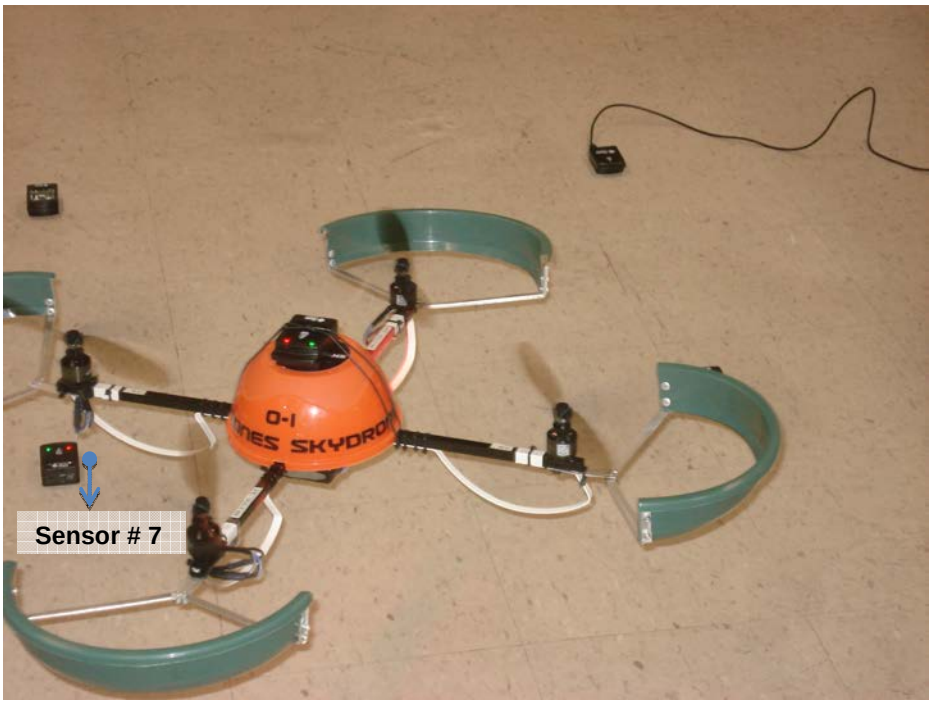


Figure 7.15: Third step of the test – after the alarm is issued, it is correct delivered by the static sensor node that is closer to the mobile sensor node (the one that has the highest pheromone level).

Figure 7.15 shows the alarm delivery from sensor node 7 (the one that has the highest pheromone level) to the sensor node carried by the quadcopter, which is noticed by the LED number 1, which is turned on with green light in both nodes.

Performance Assessment

In addition to the tests that were aimed to evaluate the appropriate operation of the network to run the trail-follow mechanism, also performance metrics were acquired from the developed demonstrator by means of measurements of the percentage of CPU time utilization, average energy resource consumption and memory usage of the sensor nodes. These performance metrics are acquired by instrumentation of the deployed code, which implied in negligible additional overhead, hence not affecting the results. In a separate control test with a SunSPOT node running just a sleeping thread and a measurement one during 100 seconds, the overhead due to the measurement thread was assessed. This assessment revealed results for the CPU utilization by the measurement thread correspondent to 0.05%, while the correspondent battery charge drain was equal to 0.214 mAh. The used memory space was of 1012 bytes.

To perform the metrics acquisition, two tests were performed. The goal of the first test was to measure the overhead due to execution of the AFME platform and the agents running on it. The second test assessed the cost to migrate an agent from one node to another. For both tests, the sensor nodes had their batteries fully charged at the beginning of the tests.

Table 7.1 presents the acquired metrics for the first test, in terms of used memory space (minimum and maximum during execution), CPU utilization, which provides the percentage of the time that the sensor node was not in sleep mode, and consumption of the battery (the energy resource), which is measured in terms of the drain in the level of the battery charge for the duration of the test, which was equal to 300 seconds. Sensor nodes with different configurations were tested, as described in the following:

Configuration 1: Control Node Platform (CNP). This configuration has an AFME platform deployed, but no agent and no services either;

Configuration 2: Control Node Service (CNS). This configuration has an AFME platform as well as all supporting services described above for the static sensor nodes, but no agent.

Configuration 3: Static sensor node with Node Agent (SNA). This is the same configuration as the above CNS, but with the deployment of a `SensorNodeAgent` agent;

Configuration 4: Static sensor node with Mobile Agent (SMA). This configuration is the same as the above SNA, but additionally is has the `AlarmAgent` deployed on it;

Configuration 5: Mobile node with node agent (UAV). This configuration is similar to above described CNS, with the AFME platform, but instead of the services deployed to the static sensor nodes, with the services designed to the mobile node. Additionally, this configuration has also the `UAVNodeAgent` deployed on it.

Table 7.1: Results for the assessed performance metrics.

Configuration	Used Memory Space (Bytes)		Battery Charge Drain (mAh)	%CPU
	Minimum	Maximum		
CNP	83704	84100	4.562	0.06
CNS	86868	87596	4.654	0.09
SNA	90648	96444	5.366	6.15
SMA	99976	112420	6.851	10.13
UAV	84892	89528	4.736	5.87

The acquired results presented in Table 7.1 show that the AFME platform by itself (CNP), and with services support (CNS) have very similar overhead. The introduction of the agents (SNA and SMA) demanded more CPU time, hence making the sensor nodes leave the sleep mode and consume more battery too. However, it is possible to observe that even with both agents deployed in a static sensor node (SMA), `SensorNodeAgent` and `AlarmAgent`, there was not a major impact in terms of memory space (around 1500 bytes more), and even with the significant increase in the percentage of the CPU utilization, the CPU was busy only a little more than 10% in SMA configuration. Noteworthy is the difference between the results for the CPU utilization of the SNA and SMA. This difference is due to the execution of the mobile agent `AlarmAgent` in SMA, which by accessing the services provided by the class `ReadBeaconServ` results in this additional overhead. The mobile sensor (UAV) has simpler platform and support services if compared to the static sensor nodes, as presented in Section 7.3. This reflects in its results, by occupying less memory space, requiring less CPU time and consuming less battery resources, compared to its equivalent configuration for the static sensor nodes (SNA).

The important information that can be extracted from the presented data is that from the deployed software point of view, the proposed trail-follow pheromone based mechanism does not imply in severe overhead when deployed in AFME in the SunSPOTs sensor nodes. It is clear that other frameworks, such as MAPS (AIELLO et al., 2011) or JADE (BELLIFEMINE et al., 2003), or even other sensor nodes, such as Mica Motes (HILL; CULLER, 2002), could be tested and perhaps provide better results. However, such a comparison analysis is out of the scope of the proposed demonstrator and the thesis. The goal of the demonstrator is only to show the feasibility of the proposed agent-based approach in a physical sensor node platform using a full featured agent-oriented platform.

The second test used two SunSPOT sensor nodes placed 75 cm apart from each other, one with configuration SNA and another with configuration SMA. The SNA node was set with a pheromone level higher than the SMA nodes, thus when the button of the SMA node is triggered, the `AlarmAgent` agent migrates from the SMA to the SNA node. Measurements are performed during the agent migration to assess the

elapsed time required for this process, the amount of bytes transferred from one node to another, and the associated drain of the battery resource. The results are presented in Table 7.2.

Table 7.2: Cost associated to the agent migration.

Measurement considering	Elapsed Time (s)	Overall Size of the Packet Frames (Bytes)			Payload (Bytes)			Battery Charge Drain (mAh)
		Sender (SMA)	Receiver (SNA)	Total	Sender (SMA)	Receiver (SNA)	Total	
Agent Transmission Only	1.000357	3378	165	3543	2619	0	2619	0.1156
Agent Transmission plus Control Messages	1.000658	3454	235	3689	2639	14	2653	

By the results presented in Table 7.2 it is possible to observe how much of the amount of data that is exchanged between the nodes corresponds to the agent itself and how much is due to control and acknowledgment messages. The effective size of that corresponding to the agent is 2619 bytes, which is the payload for the transmission of the agent only from the node SMA. Important information provided in this table is the elapsed time to perform the agent migration. It is possible to observe that there is almost no difference between the time due to the transmission of the agent and the total elapsed time (transmission of the agent plus the control messages), both being close to 1 second. This elapsed time information is important because it can be used to estimate the delay between the time instants in which an alarm is issued until it is received by a mobile sensor node, by considering the average number of hops from the alarm issuer node to the mobile one. The assessment of this delay can be used to evaluate if a given sensor node and software platform, or a combination of them, provides the required QoS for a given application. Finally, the information for the battery drain is also provided, which is presented with the same value for both transmission (of the agent only and of the agent plus the control messages), due to the inaccuracy of the SunSPOT's battery manager support to acquire values that show the difference. However, as the value itself is low, the difference should also be very small. Notice that this information by itself is not much informative, but together with the equivalent ones provided in Table 7.1, it can be used to estimate the system lifetime until the complete battery drain, for instance.

7.5 Summary

The small scale demonstrator presented in this chapter was useful to assess the feasibility of the proposed approach using software agents, supported by an agent platform running on a COTS sensor node platform. The feasibility was demonstrated by the successful deployment and execution of the trail follow mechanism, which performed the correct operation in delivering an alarm issued by a static sensor node to

a mobile one. Complementing this assessment, a quantitative analysis of the developed software was presented, which provided information about the associated costs in terms of memory space, battery resources and CPU time. The achieved results provided evidences that there was no particular concern in relation to any of the assessed measurements for the sensor node platform used in the demonstrator (the SunSPOT). The higher CPU time utilization was around 10%, thus leaving the CPU idle most of the time. The used memory space was in average less then 20% of the available space, and less than 1% of the battery charge was used by the sensor nodes to perform the alarm delivery.

8 SELECTED ASPECTS ON DEPENDABILITY IN THE SCOPE OF THE THESIS

This chapter brings a discussion about an important aspect on the topics presented in the main chapters of this thesis work, i.e. dependability. Despite that a deep discussion about dependability is beyond the scope of this thesis, it is worthy to mention and consider aspects on dependability that can affect the performance of the proposed solutions. Thus, the goal of this chapter is to provide a brief discussion about these aspects, as well as possible ways to handle them, stating that we are aware about these vulnerabilities, in spite of they are not specifically handled in scope of the presented work.

8.1 Introduction

As basic and fundamental building block of WSNs, the sensor nodes represent an obvious source of failures in the operation of WSNs. Sensor nodes can fail due a number of facts, such as exposure to harsh environments or simply energy depletion, thus influencing WSNs dependability (TAHERKORDI; TALEGHAN; SHARIFI, 2006). In order to face this problem a number of fault tolerance measures proposed in the literature explore the fact that WSNs use several sensor nodes and thus have inherent node redundancy (SOUZA; VOGT; BEIGL, 2007).

Indeed, the above mentioned idea to explore the natural redundancy provided by the great number of nodes that in general compose a WSN helps to achieve good results. Neighbour nodes can, for example, monitor each others' behaviours. When a problem is detected, such as a node crash (situation in which a node is permanently out of order), one or some of the neighbours can assume the tasks that were previously executed by the faulty node (SOUZA; VOGT; BEIGL, 2007). The assumption of redundant nodes offers the necessary condition for the success of such approaches.

However, in the lifespan of a WSN, it is probable that the node density eventually becomes lower and, fewer and fewer nodes may then be able to count on this natural redundancy, due to the failure of a number of the previously available nodes in their vicinity. Moreover, there are many situations in which it is not desired to deploy a WSN with too many redundant nodes. In these cases the overlap between the sensed areas are smaller as well as the number of alternative links to connect distinct nodes, thus reducing the robustness of the design. An example of such situations in which redundancy based on a unnecessary high density of nodes in not desired is the

deployment in an area in which the nodes should be as invisible as possible, e.g. for secrecy, and in which a high concentration of them would offer an opportunity to easily detect their presence.

Moreover, besides the sensor nodes themselves, the network among them may also present problems that are not directly related to the individual nodes, but have other sources, such as the environment where they are deployed. Thus, before studying the main vulnerabilities to which the proposed solutions in this work are subjected to, it is important to understand the cause-effect relation applicable to dependability problems of WSN in a broader sense.

8.2 Dependability Issues in WSN

The notion of dependability is broken down into six elementary component properties (AVIZIENIS; LAPRIE; RANDELL, 2001) as follows:

a) Availability: the system will be operational when needed;
b) Reliability: the system will keep operating correctly while being used;
c) Safety: the system operation will not be dangerous;
d) Confidentiality: there will not be disclosure information when not authorized;
e) Integrity: there will be no unauthorized modification of the information used by the system; and
f) Maintainability: the capability to perform a successful repair action within a given time.

Considering WSN, the two first aspects of dependability are a must (TAHERKORDI; TALEGHAN; SHARIFI, 2006). It does not mean that the other ones are not important, but these two are the ones that concentrate the focus of attention in the research community for a long time, as discussed in (TAHERKORDI; TALEGHAN; SHARIFI, 2006). Some of the others aspects correspond to complete own research areas by themselves, which is the case for confidentiality and integrity. These two topics are focused by the research on security solutions for WSNs (WANG; ATTEBURY; RAMAMURTHY, 2006). On the other hand, maintainability can also be considered as a hot topic in WSN research. Several proposals aim at reprogramming the entire network, which can be seen as a way to fix or evolve the software pre-installed in the sensor nodes, such as (LEVIS; CULLER, 2002), in which a framework for sensor nodes software update is supported by a virtual machine installed in the sensor nodes. Even the mission dissemination approach presented in this thesis can be considered as a contribution if instead of specifying sensing missions, the software agents execute update of software components that might be corrupted, deprecated or obsolete. However, this possibility should be observed as a possible “side-effect” of the approach, as dependability is not focused in this work, and neither in (LEVIS; CULLER, 2002).

Most of WSN applications should run continuously and correctly during their operation. They are characterized by being mission critical, complex, composed of components that have significantly fault rates, such as semiconductor integrated circuits-based systems and sensor devices, besides their software components that run the WSN applications (KOUSHANFAR; POTKONJAK; SANGIOVANNI-VINCENTELLI, 2002). All these aspects together reveal a big threat to WSN operation,

deriving faults that trigger the rest of the error-failure-fault-error… chain compromising the system functionality (AVIZIENIS; LAPRIE; RANDELL, 2001). These problems affect the WSN system availability and reliability, supporting the strong interest and focus on these two topics and the need for understanding the fault-error-failure chain in WSNs.

8.2.1 Faults

Considering just the network of sensor nodes, and discarding the issues related to systems connected to it, such as the backend information systems that provide user interface to the WSN, the source of faults in WSN can be then classified as: 1) Node Faults and 2) Network Faults (SOUZA; VOGT; BEIGL, 2007).

Node Faults: Node faults are considered to be internal faults that occur in individual sensor nodes. These faults may have their cause in the hardware or in the software components of the nodes. In the hardware, common problems are related to antennas that are usually fragile and may not resist to impacts (LANGENDOEN; BAGGIO; VISSER, 2006) and to batteries that provides incorrect voltage or current outputs damaging other internal components (TOLLE et al., 2005).

Despite that hardware faults represent an important source of problems, software related faults are very common and may exceed the first ones, depending on the complexity of the WSN application (SOUZA; VOGT; BEIGL, 2007). As deeply embedded systems, sensor nodes are hard to program due to the inherent resource constraints of such systems. Incautious programming of sensor nodes may easily lead to memory overruns, buffer overflows, deadlocks, all sorts of tricks and hard to debug programming problems. Such faults may cause incorrect data handling, an error, which can manifest itself as a failure by providing an erroneous result to other nodes. This failure may trigger a fault in the data aggregation software in other nodes, as the WSN in general run distributed applications.

Network Faults: Network faults are related to the interaction among nodes and lead to errors that affects the nodes' communication and coordination, as in the data aggregation problem mentioned above. One of the very common sources of faults is radio wave interference. This problem may occur by obstacles present in the environment, or by the presence of wave reflecting surfaces or other kinds of devices or networks operating in similar frequencies, or even by other nodes of the same network placed close by. In fact this last case, even a legitimate emission by a neighbour node may cause undesired interferences especially at the limits of a nodes range when it receives low signal strength. Link stability is also reported as a problem in sensor nodes networking (SOUZA; VOGT; BEIGL, 2007). This instability affects the sensor nodes routing capabilities, which per se is another source of problems. This issue affects specially single-hop solutions, which are more vulnerable in relation to this aspect than multi-hop alternatives.

Routing issues can also be considered as a significant source of networking malfunctioning. In WSN, routing has a great importance which is specially related to the semantics of the applications that run over the networks (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). Faults in the routing protocols may compromise the network as a whole, or parts of it, in a number of ways (e.g. the wrong

cluster-head election in hierarchically organized WSN due to delayed, dropped or misrouted messages).

It is important to observe that this classification complies with the fundamental concepts in dependability presented in (AVIZIENIS; LAPRIE; RANDELL, 2001). For example, the software faults discussed above, such as buffer overflow, can be classified as: Development; Internal; Human-made; Software; Non-Malicious. These classes roughly translate to: caused in design or development process, on which an internal part of the system, a software part was programmed by a person that did not consider checking the input value, so it was human-made introduced, and not necessarily malicious. It could however be malicious if the programmer intentionally considered the hypothesis of such condition to happen.

8.2.2 Errors

Errors are deviations of the normal way that a part of a system should work, and they are activated by faults (AVIZIENIS; LAPRIE; RANDELL, 2001). The types of errors considered in this section follow the consequences of the faults presented above as exemplified by miscalculation errors due to programming faults (SOUZA; VOGT; BEIGL, 2007).

Another example of an error is caused by a calculation or output of a value based on wrong data provided by a faulty sensor device. The sensor device may provide an unexpected value, which is out of the range of tractable values for the piece of software that manipulates this data. As a result, an erroneous output is generated being a manifestation of two original faults: the former is a hardware fault, related to the sensor device that provided the wrong value; and the latter is the fault due to lack of protection in the program code against out of bound values, which leads to an erroneous state.

Errors due to network faults can be exemplified by selection of wrong routes to forward messages due to routing information faults. The consequence of these faults can be erroneous data aggregation, for example, or undelivered expected messages. Malformed or incomplete communication packets can be presented as examples of errors due to interferences.

8.2.3 Failures

A failure represents an incorrect delivery of a service provided by a system. Originally motivated by one or more faults, it is the external manifestation of an error.

In WSN a failure could be considered the final erroneous information provided to the end user, which is ultimately the desired service from a WSN. However, for the scope of this thesis work, the considered failures are related to the nodes and the network, as mentioned above and classified according to (SOUZA; VOGT; BEIGL, 2007):

Crash or Omission Failures: Failures in which the nodes or parts of the network stop to respond requests. A sensor node may crash due to a permanent fault in its sensor or transceiver device, or due to the exhaustion of its battery. A region of a network may stop responding due to interferences that hinder the communication in that region or due to isolation in case of crash failure of the intermediate nodes that enable communication with a region.

Timing Failures: Failures that occur due to mismatch between the timing requirements specified for the WSN services and the actual observed timing behaviour. A service may deliver a correct value, but if it is out of the predefined timing bounds, it is considered a timing failure.

Value or Semantic Failures: Failures in the form of deviations in the expected outputs of the system in terms of the content of data or information. An example of such failure can be the misinterpretation of an alarm sent by a ground sensor node leading an UAV to an incorrect location, in the case of the approach to address the cooperation among static and mobile sensor nodes presented in Chapter 5. Another example is the miscalculation of an aggregated value from a set of sensors.

Arbitrary or Byzantine Failures: Failures that do not follow a consistent pattern. An example of this is a service that delivers intermittent either correct or erroneous outputs. This behaviour makes it hard to detect the source of the failure. Intentionally inserted malicious faults may lead to errors that express themselves as byzantine failures.

8.3 Dependability Issues in the Proposed Solutions

In light of the summary description of dependability issues in WSNs presented above, it is possible to discuss the main problems that the approaches proposed in this thesis may face that can hinder their proper operation. Despite the specific problems that are discussed separately for each solution in the following, an issue that is common for them three is related to the reliability in the message delivery, which is out of the scope of the thesis, and is the reason why the evaluation of the proposed solutions is expressed by the cost in terms of messages that are sent instead of the messages that are received by the nodes.

8.3.1 Sensing Mission Dissemination and Allocation in Static WSN

Considering the approach presented in Chapter 3 to address sensing mission dissemination and allocation, there are two main issues related to dependability that can present negative effects, namely gaps in the network connectivity and premature start of the mission allocation decision procedure.

The first issue directly affects the mission dissemination phase of the proposed solution. If one or more nodes of a network are disconnected, i.e. if there are nodes that do not have links to the rest of the network and thus are not possible to reach, these nodes will not receive the missionAgents during the mission dissemination phase. The communication failure due to such network disconnectivity may have been caused by a crash or failure of one or more nodes that provide the linkage between a node or group of nodes to the rest of the network.

The failure in the mission dissemination as described above can be responsible for a malfunctioning behaviour of the mission allocation procedure, resulting in an error, in providing wrong evaluation of the nodes' neighbourhood, in its turn providing a failure resulting in poor performance in the allocation of missions to sensor nodes.

Because of the many different kinds of problems that the disconnectivity can induce, the evaluation of the proposed approach presented in Section 3.4 assumed that the nodes distributions used in the experiments form connected networks.

The second named issue is referred to the way the mission allocation procedure is performed, which relies on a timer to start its activation. The problem in this case is related to the premature start of the allocation procedure (error) due to erroneous timer expirations (fault). As a result, a poor allocation may result in a failure due to the fact that a node did not have time to receive information about its neighbours.

To face the communication issue related to the gaps in the network connectivity, the deployment of redundant sensor nodes can provide alternatives links to keep the network connected. However, in the long run the connectivity problem will eventually appear when the WSN comes closer to the end of its lifetime.

A measure to overcome the issue related to the time requirements associated with the start of the allocation procedure is to the use of larger time intervals to configure the timers, so that it is guaranteed that the mission dissemination is finished. However, physical problems in the timers are still possible to happen, which may provide erroneous values, resulting in premature start of the procedure anyway.

8.3.2 Sensing Mission Dissemination in Mobile WSN

Observing the approach presented in Chapter 4, besides the problems related to the establishment of communication links between mobile nodes, which is a special problem in VANETs as discussed in (LIDSTRÖM; LARSSON, 2010), especially when single-hop communication is considered, the major issues that may affect the presented agent migration strategies are incorrect information that support their decisions to switch nodes, and the node density.

As explained in Section 4.2, the missionAgents' decisions to move among the nodes depend on the information that they receive from the nodeAgents. If this information is wrong or imprecise, wrong decisions can be taken thus degrading the overall system performance. A fault that can be named as cause for this kind of problem, is a high imprecision in the positioning information provided by a GPS, which leads to an erroneous evaluation if a position is inside or outside the MA, manifesting a failure in taking the correct decision to move or not.

As already discussed in the Section 1.2.2, node density is an important aspect in the application scenario. Observing the proposed strategies presented in Section 4.2 and their respective reported results in Section 4.3, the low density of nodes negatively affects specially the strategies that use less context information. However, even those strategies that use more context information are also affected, even in a minor degree.

8.3.3 Alarm Handling in Cooperative Static and Mobile WSN

The combined use of static and mobile sensor nodes is subject to a number of problems that may hinder the smooth and correct operation of the network. The major issues to be considered here are related to the network connectivity among the static nodes on the ground, which affects also the mission dissemination as discussed in Section 8.3.1, and may result in gaps in the sensing coverage.

This thesis work considers that the static sensor nodes are randomly deployed according an independent uniform probability distribution (homogeneous Poisson point process in two dimensions, which generates a geometrical random graph). The condition to guarantee that a graph generated by nodes distributed in such a way forms a

connected network is assuring that for each node, there is at least one link that connects it to the rest of the network, which has its probability calculated by:

$$P(d_{\min} > 0) = (1 - e^{-\rho r^2})^n \ . \quad (8.1)$$

where $P(d_{min} > 0)$ is the probability that the minimum degree of the network is higher than zero, considering the above described distribution conditions, "ρ" is the node density, "r" is the communication range, and "n" is the total number of nodes in the network (BETTSTETTER, 2002).

For the simulations presented in Section 5.6, the number of sensor nodes employed resulted in a connectivity probability close to 100% (>99.999%), for all tested sizes of areas. This is a necessary condition to assure that the alarm forwarding mechanism does not get stuck due to gaps in the network connectivity.

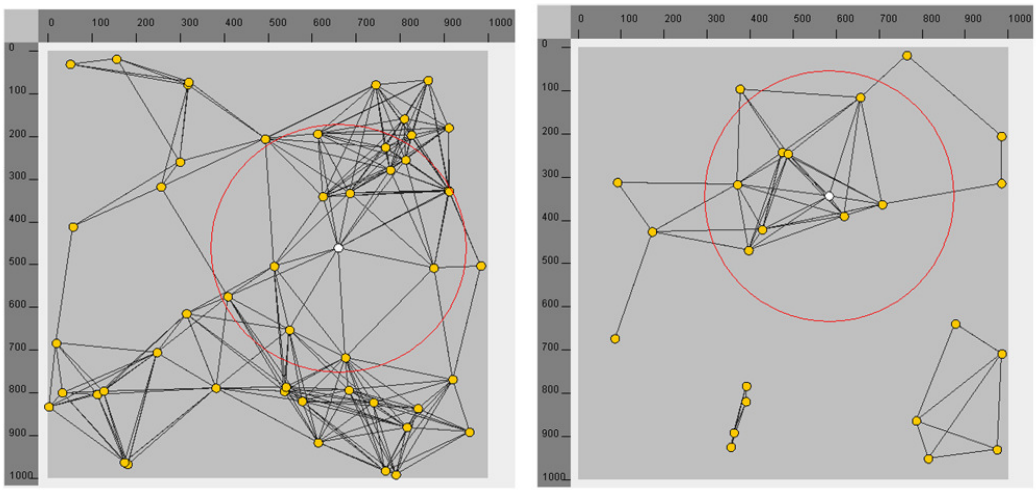
However, during the WSN life time, nodes have their energy depleted and crash failures occur. With the increase in the number of node crash failures, the probability that the network forms a connected graph drops and that the probability of gaps in the connectivity increases, giving rise to a network failure. Table 8.1 presents the values for the node connectivity of a network deployed in a squared area with sides of 10 Km length, in which sensor nodes with 350 meters communication range are randomly distributed according to homogeneous Poisson point process in two dimensions. According to (8.1), a number equal to 5000 nodes provides $P(d_{min} > 0)$ approximately equal to 1. Starting from this ideal number of nodes, the table presents decreasing number of nodes and their corresponding values for $P(d_{min} > 0)$.

Table 8.1: Number of nodes and their respective $P(d_{min} > 0)$.

Number of Nodes	% of the reference (5000)	$P(d_{min} > 0)$
5000	100	~1.0
4500	90	0.999863268
4000	80	0.999168572
3500	70	0.995031912
3000	60	0.971209180
2500	50	0.846539727
2250	45	0,675534269
2100	42	0,521033789
2000	40	0.401629927
1500	30	0.009203929

Notice that below 2500 nodes (50% of the ideal number of nodes), the probability that the network is connected drops very fast. This happens because the nodes become

more sparsely distributed, thus creating more possibilities for gaps in the network. These gaps compromise the alarm forwarding mechanism that cannot guarantee the alarm delivery. To give an idea of the problem generated by gaps in the network connectivity, Figure 8.1 shows two areas of 1 Km^2 in which nodes having communication range of 350 meters are randomly distributed according to the conditions mentioned above. In this figure the nodes are represented by small circles connected by lines that represent the communication links among them. The big circle that has a white coloured node in the centre shows the communication range of a node, each node is assumed to have similar range but as not to clutter the figure just one is shown.



(a) (b)

Figure 8.1: (a) Connected graph with 50 nodes; (b) Disconnected graph with 25 nodes.

In Figure 8.1a it is possible to see a network formed by 50 nodes in a 1Km^2 area, thus the same node density for the ideal number of nodes presented in the first row of Table 8.1 (50 node/Km^2), which provides a result for $\boldsymbol{P(d_{min} > 0)}$ very close to 1. Figure 8.1b shows the situation of a disconnected graph with 25 nodes (50% of the ideal number of nodes). It is possible for network consisting of 25 nodes to be fully connected following the distribution conditions presented above. However, as shown in the example of the Figure 8.1b, the network is very sparse and not always very uniform, and in this example the network is disconnected having three partitions. Under such conditions, alarms issued or forwarded by nodes in one of these network partitions stay stuck and do not manage to proceed following a trail. For this reason it is proposed in Section 5.3 that in such cases in which an alarm reaches a stuck situation while following a trail (trail-follow), it starts performing a trail-search again. Even performing a trail-search, it is possible that while following a given direction the alarm will get stuck, in such situation it selects a different direction before continuing with the trail-search. However, this state of network disconnection is an undesired situation and the proposed use of the trail-search to circumvent such problem is intended to make the system more robust against it. The ideal is to have the network fully connected as considered in the performed experiments reported in Section 5.5.

For faults in the sensor nodes that result in sensing failures, the problem is the absence of sensing coverage in parts of the monitored area. This lack of coverage is a system fault that will manifest a failure in detecting possible threats. In cases in which the threats are mobile entities, such as vehicles, this problem is softened by the fact that they will eventually be detected by other sensor nodes that are operating correctly, while the threats move through the monitored area. In the case of static threats this might however cause bigger problems. For instance, a fire is static in the sense of its origin, but it spreads affecting surrounding areas from its starting spot. As it spreads, larger portions of an area are affected, and like this it will be eventually detected by sensor nodes located in these surrounding and also affected areas. However, depending on the lack of coverage left by the faulty node(s), the fire might have become too big and the benefit of using a WSN to monitor the occurrence of a fire spot may be lost.

Both issues described above are able to be handled by using redundant sensor nodes or deployment of new sensor nodes in areas were faulty ones are detected. To do this, it is necessary to adopt solutions for fault detection, such as group detection techniques as the one presented in (CHEN; KHER; SOMANI, 2006). In this type of solution, the sensor nodes keep monitoring their neighbours so that abnormal behaviours are detected to identify faulty nodes. Once the number of faulty nodes exceeds a given threshold that characterises gaps in the network connectivity or sensing coverage, new sensor nodes can be deployed. Another possibility is the use of the mobile sensor nodes to provide a backup support to cover areas with sensing gaps. Using this approach, gap areas can be defined as preferred ones to be visited more often by the mobile nodes than other areas. This approach is presented in (FREITAS et al., 2011b), in which the adoption of different policies to guide the movement of the mobile nodes to cover gaps in the sensing coverage is discussed.

8.4 Summary

The discussion presented above highlights the main problems that can affect the correct operation of the solutions proposed in this thesis work, regarding the assumptions made in Chapters 3, 4 and 5. These assumptions are mainly related to the network node density, connectivity and appropriate sensing operation.

As exemplified in Section 8.2, there are several problems related to dependability that may affect WSNs. The provided examples are not exhaustive, as there are many others that are reported in the literature, as presented in (TAHERKORDI; TALEGHAN; SHARIFI, 2006). Many of these other problems may affect the operation of our proposed solutions too, but they are common to any WSN that are not specifically designed to address them.

The primary goal of the presented analysis is to state the awareness of the main dependability vulnerabilities that may affect the proposed solutions, taking into account the considered assumptions, and how they can be handled to decrease their impact. It is noteworthy to mention that a deeper analysis on dependability aspects on WSNs is out of the scope of this thesis work. It has no ambition to provide an exhaustive description of the possible problems, but only an overview of such aspects and this to give a context

to the presented analysis of the proposed solutions. A secondary goal is to highlight future works that address these dependability issues to enhance our proposed solutions.

9 RELATED WORKS

The solutions proposed in this thesis to address the three selected problems in WSN-based surveillance systems described in Section 1.4, have relation with several different works in the WSN research area, depending on these specific problems. Keeping the organization of the problems' presentation and their proposed solutions as in previous chapters, selected related works are grouped in this chapter following the same order.

As it is discussed in the following, some of the described related works present approaches to address the problems described in Section 1.4, while others focus on different research questions, but which are still related to the scope of this thesis. The importance in mentioning these works is that, despite the difference in the specific addressed problem, they employ techniques that have important similarities and share similar application context as presented in our work. Additionally, many of the related works have complementary features that can be jointly used with our proposals, enhancing the expected results of one another.

9.1 Sensing Mission Dissemination and Allocation in Static WSN

Agilla

Agilla (FOK; ROMAN; LU, 2005) and its evolution (FOK; ROMAN; LU, 2009) present an approach that uses mobile software agents able to setup the sensor nodes according to data availability and locality. The mobile agents can migrate and clone to disseminate themselves in the network. Agents hosted by one node coordinate with other agents hosted in neighbour nodes by using tuple space techniques (GELERNTER, 1985) to share data, instead of message exchange directly among the agents.

Our proposed approach for mission dissemination in static WSN is closely related to Agilla, as it explores the mobile agent capabilities of migration and cloning, but an important difference is in the decisions taken by their agents in relation to ours. In their work, the agents move around the network trying to find sensed samples that match a given criterion and, based on that, provide data from those nodes to the base station. This approach may lead to an inefficient engagement of sensor nodes, since, to provide certain information, more than the necessary nodes may be used due to the positive match with the established criterion. In comparison, our strategy tries to use sensor nodes in a more rational way, keeping the number of engaged nodes as close as possible to the desired amount of nodes needed to perform a given sensing mission, according to the mission directions.

SAMSON

In (MORRIS; GIORGINI; ABDEL-NABY, 2009) the authors present a BDI-agent (Belief – Desire – Intention) model and use it to simulate the operation of WSNs. The agents representing the sensor nodes are able to decide about the actions that the nodes have to perform in order to accomplish the specified sensing tasks. The strength of this work is in the detailed BDI model for the agents that represent the sensor nodes, which is used in the SAMSON WSN simulation tool. SAMSON includes models for the sensor nodes' hardware and also models of the operation environment.

Despite the authors' contribution in modelling a WSN using a complete agent model, AgentSpeak language (RAO, 1996) and Jason framework for agents (BORDINI; HUBNER; WOOLDRIDGE, 2007), the authors do not spend much on the reasoning mechanism itself, as it is presented in our approach described in Chapter 3. It is possible to state that their work and ours are to a certain degree complementary. Their approach could be incorporated into our proposal to model the agents, providing possibilities to extend our approach with a more sophisticated data manipulation, while our decision mechanism would enhance the intelligent capabilities of their agents.

FLOODNET

The authors in (KHO; ROGERS; JENNINGS, 2009) present the employment of agents to provide distributed intelligence to sensor nodes in the FLOODNET project. The FLOODNET project designs a flood prediction system based on sampled data about the water level in a flood collected by a set of sensor nodes. Due to the distribution of the sensor nodes, frequent maintenance to change batteries is not practical. Hence, a solution that keeps the network providing useful data and uses the energy resources in a rational way to extend the batteries lifetime is desired.

To achieve such an energy aware sampling, their work is focused on the development of an agent-based solution to provide adaptive sensing or sampling capabilities to the sensor nodes. Their proposal uses information metric that grades the value of gathering the data at a certain frequency, considering the actual conditions that the sensor node is experiencing. This is done by using a Gaussian process regression to infer the characteristics and the value of a function that represents the metric. Based on this achieved value, three different decentralized control algorithms are used to perform the adaptive sampling by means of a trade-off between cost and optimality.

Their work assumes that the sensing missions are already distributed and assigned to the sensor nodes, while our approach addresses this problem as described in Chapter 3. On the other hand, they provide mechanisms to adapt the behaviour of the nodes the mission needs during its execution, which is beyond the scope of our proposal. However, it is important to note that both works have complementary properties, as a mission disseminated and assigned according to our approach could have its performance adapted by means of their mechanisms. On the other hand, the use of our approach to disseminate and allocate sensing missions in their network could provide a reduction in the number of sensor nodes engaged in performing the mission, thus resulting in energy savings from the beginning of the mission accomplishment.

TEDD

A protocol called Trajectory and Energy-based Data Dissemination (TEDD) is presented in [MGM05]. TEDD combines concepts of trajectory-based forwarding (TBF) (NICULESCU; NATH, 2003b) with energy maps (MINI et al., 2005), which provides information about the nodes' energy reserves, to dynamically determine routes by which messages will be forwarded to disseminate information in WSN. While a message follows a given trajectory, only those nodes with higher levels of energy that are close to the pre-determined trajectory forward the message. Nodes with lower levels of energy avoid forwarding the message, saving their energy to sensing tasks and to receive disseminated messages.

The TBF approach in their work may be compared to the geo-awareness of our approach in the mission dissemination phase presented in Section 3.2. The energy map evaluation is not used in our approach for the mission dissemination itself, but only in the mission assignment, in cases in which the energy level is one of the parameters of the goodness function. An extension of the algorithm described by the flowchart presented in Figure 3.5 can include the consideration about energy in addition to the geographic condition. An important difference between our approach and their work is that they try to minimize the energy consumption of message dissemination from a sink node to the sensor nodes only. However, in our approach we are not only concerned with the delivery of the message coming from the sink, i.e. the mission, but also in how to make use of this communication to exchange information among the sensor nodes, avoiding additional communication to perform the mission assignment.

Unified Broadcast

Unified Broadcast (HANSEN; JURDAK; KUSY, 2011) presents an approach to combine the broadcast messages requested by different protocols in WSN in a single broadcast, while maintaining a modular architecture of the network stack. According to the authors' proposal, data from protocols that demand flooding messages such as time synchronization, query dissemination and network status monitoring can combine their broadcast messages so that instead of sending one broadcast for each of them, the sensor nodes combine their information and send them all together. To realize this proposal, the authors implement a transparent layer that is placed between the network and the link layers. This layer receives the packets from the network layer dispatched by different upper layer protocols, and then delays and schedules them so that they can be combined in one message that is sent to the link layer and further transmitted as a broadcast message via the wireless channel.

The goal of this work is the same as ours in the sense of providing a reduction in the number of broadcast messages sent by the sensor nodes. In our case, missionAgents collect information from the sensor nodes while the mission is being disseminated, which will then be used in the mission allocation procedure. If this information was not piggybacked during the mission dissemination, the nodes would have to send another broadcast informing their neighbour nodes about their conditions. While our approach is more specific for the mission dissemination and allocation problem, their proposal is more generic and can be used for other protocols. However, their approach is more suitable for services that need to send periodic broadcast messages, such as time synchronization, while ours better addresses broadcasts of aperiodic nature, such as the dissemination of new sensing missions.

WMOS

In (LI; ZHAI; SUN, 2010) the authors propose a publish/subscribe middleware to support sensing missions dissemination and information retrieval in wireless sensor networks. In their proposal the users enter queries in form of topics of interest, which are then processed by the front-end of the middleware and translated in a contented model, depending on the QoS requirements. After this process, the queries are sent out to the network in XML format by an ordinary flooding mechanism. When receiving queries, the sensor nodes perform a matching, telling what they are able to publish and the subscription requirements specified in the incoming XML. When an event that matches a subscription happens, the nodes inform the sink node via multi-hop messages. The sink node then collects this information and sends it to the middleware that presents the information to the subscriber (final user).

The idea of the matching between the sensing requirements described in the XML queries proposed by the authors and our mission specification that is used by the missionAgents to select eligible nodes for a given sensing mission is similar. However, in their approach all sensor nodes are engaged in a given sensing mission, while in ours there is a mission allocation procedure applied after that the mission is disseminated. This mission allocation procedure selects among the eligible nodes a subset that will indeed engage in performing the mission. Our approach reduces then the number of nodes that will report the same information to the sink (redundant nodes), while their approach does not take this into account. Even though data aggregation is not within the scope of this thesis, the effect of our approach corroborates with the goals of a data aggregation strategy, as it can diminish the amount of redundant information that is sent back to the sink. On the other hand, their approach needs a specific data aggregation process to perform this reduction; otherwise all redundant information will reach the sink node. Moreover, their mission dissemination process is based on ordinary flooding, while ours is based on a less costly method in which the sensor nodes avoid sending redundant messages by the clone/migration-clone/discard decisions taken by the missionAgents.

A Decentralized, On-line Coordination Mechanism for Monitoring Spatial Phenomena with Mobile Sensors (DOCMSP)

In (STRANDERS; ROGERS; JENNINGS, 2008) an on-line decentralized mechanism is introduced for coordinating multiple sensors in order to monitor an environment modelled using Gaussian Process (GP). This work represents an example of an implicit cooperation, in which agents reach a common picture of the environment, but decisions are taken locally (without consulting other agents) (VINYALS; RODRIGUEZ-AGUILAR; CERQUIDES, 2011). This makes their work comparable to ours, as the decision-making procedure carried out by the agents to allocate missions in our work is also made locally, without additional message exchange among the agents to agree about the mission allocation. Indeed, this is one of the important factors that enable the reduced overhead presented by our approach to assign missions to the sensor nodes, as shown in the results presented in Section 3.4. The difference between this part of our work and theirs, besides the different methods to carry out decisions, is that their work is focused on driving mobile sensors towards areas that can provide more valuable

measurements while our approach presented in Chapter 3 addresses the setup of static sensors to allocate mission according to their specific requirements. However, in spite of the fact that we also present solutions for WSN consisting of mobile nodes, the approach described in (STRANDERS; ROGERS; JENNINGS, 2008) is comparable to the mission allocation solution that we presented for static sensor nodes in Chapter 3. This is due to the similarity in the way both works explore local information to form a common picture of the network in each sensor node. Thus it is more related to our proposal to address the setup of static WSN presented in Chapter 3 than to our approach in addressing mobile sensor nodes.

Summary Comparison

To complement the above presented analysis, Table 9.1 presents a summary comparison including the proposed approach and the selected related work. The following criteria are evaluated in this summary comparison:

a) Node Heterogeneity: evaluates if the approach considers sensor networks composed of nodes with different capabilities;
b) Adaptiveness: assess if an approach addresses the problem related to change the way a sensor node is performing a sensing task during runtime. Some approaches may not directly address this issue, but can present possibilities to do so by extensions or combinations with other approaches. An approach is not addressing adaptiveness if such possibilities for extensions or combinations are not clearly noticeable;
c) Cooperation Among Nodes: nodes in a WSN may cooperate explicitly or implicitly. The former explores protocols such as negotiations among nodes while the latter explores local decisions taken by the nodes that at the end reach common goals;
d) Synchronization Dependent: evaluates if a given approach or its applicability is dependent to time synchronization among the sensor nodes;
e) Test / Validation: provides information about the way an approach was tested;
f) Tools: it reports the tools used to perform the tests.

Table 9.1: Summary Comparison of the Analysed Mission Dissemination in Static WSN Related Work.

Approaches	Node Heterogeneity	Adaptiveness	Cooperation Among Nodes	Synchronization Dependent	Test / Validation	Tools
Our Approach	Considered	Possible	Implicit	No	Simulation	ShoX and GrubiX
Agilla	Not Considered	Possible	Implicit	No	Prototype	Mica2 Motes and TinyOS
DOCMSP	Considered	Addressed	Implicit	No	Simulation	Custom Environment
FloodNet	Not Considered	Addressed	Implicit	Partially	Prototype	Custom Environment
SAMSON	Considered	Possible	Explicit/ Implicit	No	Simulation	Jason
TEDD	Not Considered	Addressed	Implicit	No	Simulation	ns-2
Unified Broadcast	Not Considered	Addressed	None	Yes	Simulation	TOSSIM
WMOS	Considered	Not Addressed	None	No	Not Mentioned	Not Mentioned

9.2 Sensing Mission Dissemination in Mobile WSN

GBMA

The Geographically Bound Mobile Agent (GBMA) proposed in (TEI et al., 2005) have similar goals as our solution presented in Chapter 4, in which an agent is sent to a given area to collect data and migrate from node to node of a MANET to stay in that area. In GBMA the agents use an assigned region (called required zone) where they perform the migration to other nodes in an attempt to reach nodes inside the area of interest (called expected zone), which corresponds to the mission area in our proposal. When an agent detects that it is in the required zone, it tries to find a route to reach the expected zone that can be provided by one of its neighbour nodes. If available, one of these neighbour nodes provides a route to a node inside the expected zone. This route can be single or multi-hop. Once it succeeds in finding such a route, it starts the migration to reach that node.

Differently from their proposal, in our approach presented in Chapter 4, the agents do not have this constraint in terms of a specific zone in which they have to perform the migration. The only condition that is taken into account is that the agent's hosting node be outside the mission area. When the agent detects such a situation, it tries to migrate to other nodes that have higher probability to take it to the mission area. Hence, our approach seems to be more robust than the one proposed in GBMA, especially in cases of low node density in which there is probability that the nodes moving outside the mission area do not meet any other node for a while. In such cases, there are risks that the nodes come outside the bounds of the required zone, which means that the agents may loose the opportunity to migrate, while we do not have such limitation in our proposal. Moreover, they do not consider different levels of intelligence to decide about the agent migration, as considered in our work. However, they address multi-hoping, while our approach is based on single-hop.

Vehicular Networks in Urban Transportation Systems (VNUTS)

A mobility-centric approach for vehicular networks is presented in (WU et al., 2005), which has its results for the data dissemination analysed in (WU et al., 2009). This work tries to combine the ideas of opportunistic, trajectory based and geographical forwarding in a unified proposal. The authors propose the use of inter-vehicular communication to disseminate data among the nodes of a vehicular network, and to provide message relaying to a source and destination nodes that can be static or mobile. They also consider the use of road-side communication units to assist in relaying messages among the vehicles when the density of vehicles is too low. The authors study the influence of the mobility in the data traffic, presenting results in different scenarios with varying node densities and vehicle speeds.

Their concepts are similar to ours presented in Chapter 4, i.e. opportunistic networking, and geographical awareness. However, they do not explore the geographical information as much as we do neither the different levels of geographical information as presented in Chapter 4. Rather, they just use the information about the direction of the nodes. This difference between the two works makes their approach more suitable to communications that have a precise destination, thus nodes moving in the direction of the destination can be selected as forwarding nodes. On the other hand, our approach does not have a precise node as destination, but an area (the mission area). Any node in this area can be the destination. Observing this condition, their approach can be considered one possible decision criterion for the choice of the forwarding nodes,

but it should be adapted in order to abstract the issue about the destination node. Another important difference is that they consider the possibility of static nodes (the road-side units) to assist the vehicles in relaying communication, which is not a valid assumption in our work.

SHAFT

In (MEYER; HUMMEL, 2009), the authors present a protocol to disseminate data in MANETs using an opportunistic approach based on analysis of geo-location of the nodes. In this approach, the data is associated to a geographical position, which they call a Point of Interest (POI), and the mobile nodes replicate the data in a circular area around the POI. They avoid unnecessary communication overhead by limiting the number of mobile nodes that forward a data within this circular region around the POI, which is divided in sectors and from this division comes the name SHAFT (Sector Heads Aided Flooding Technique). The nodes' geographical placement in relation to the sector division is used to select those that will be used in the data forwarding.

A common aspect shared between SHAFT and our approach for sensing mission dissemination in mobile wireless sensor networks is the exploration of the geo-context information to make decisions about whether to communicate or not in a MANET. However, differently from their proposal, ours does not broadcast data inside a certain area to all the nodes in the network, but instead move data (in our case the missionAgents) across the nodes to reach a specific location (the mission area). While their technique is closer to flooding, ours is closer to routing.

Mobile Agent as an Approach to Improve QoS in Vehicular Ad Hoc Network (MAQoS)

In (KUMAR; DAVE, 2010) the authors survey the applicability of mobile agents to improve QoS features in applications running on VANETs. Details about the main characteristics of VANETs and the key features of using mobile agents are presented, making a logical link on how to use these features to support QoS in VANET applications. A sketch of an agent-based solution is presented, in which different classes of agents are designed. These agents have different tasks and migrate around the nodes according to them.

The multi-agent system proposed in their work has some similarity with our approach, as the presence of a static agent in each mobile node (the nodeAgent in our work), which provides information about the node to the mobile agents temporally hosted by the node. However, the application of their mobile agents goes beyond of the scope of our work, and allows for instance the discovery of multi-hop routes between two specific nodes. Considering the high dynamicity of the nodes in a VANET, this approach may imply high costs to maintain such routing information up to date, which is against the goals of our approach. While a plus of our work is the use of geographic information to take intelligent decisions about when an agent should migrate or not, the authors in (KUMAR; DAVE, 2010) do not explain how the decisions about the migrations of their agents are performed.

Mobeyes

Mobeyes (LEE et al., 2009b) is a project developed at UCLA that proposes the use of a Vehicular Sensor Network (VSN) to monitor urban environments against the action

of criminals and terrorist attacks. To implement an application with this purpose, the use of static sensor nodes is considered vulnerable and hard to scale. The vulnerability comes from the fact that such attackers would be aware about the existence of these countermeasures and would try to neutralize them before performing the attacks. The scalability problem comes from the fact that such a system need to provide a wide coverage, for example to cover major areas of big cities such as London or Madrid, which would imply in prohibitive costs. Besides, the amount of information provided by such networks of statics sensor nodes would require a big infrastructure to process and analyse them. Thus the need for a decentralized and stealthy solution is clear.

The alternative presented in (LEE et al., 2009b) proposes a VSN in which cars equipped with communication and sensing devices can acquire and analyse data from video and other sensor data to extract features that indicate possible threats. These indications would then be collected by cars or other equipment from the police officials to take the necessary actions. This communication occurs in an opportunistic fashion in which the cars use their temporary neighbours in the vicinity to route their messages. These messages have a reduced size due to the local processing of the acquired raw data, allowing a rational used of the communication media.

Mobeyes uses mobile software agents to collect summary information from the processed raw data provided by the vehicles. The similarity with our approach is the use of geographic context information besides the information semantics to decide about the communication performed by the data collector agents in Mobeyes. However, in Mobeyes the agents are used to collect the results of a sensing mission, while in ours the agents are used to disseminate a sensing mission in the network. From this perspective our proposal, presented in Chapter 4, can be seen as complementary to their work as it provides a flexible way to setup the network for new sensing missions, while they assume that a sensing mission is already setup from the beginning of the system runtime.

Summary Comparison

Table 9.2 complements the above analysis by a comparison tanking into account the following criteria:

a) Node Density: evaluates if the approach considers effects of the node density in its results;
b) Mobility Model: informs the different mobility models used by each approach (Manhattan Mobility Model, Random Waypoint Mobility Model, Freeway Mobility Model (BAI; HELMY, 2006) or Real-Track (ZHOU; XU; GERLA, 2004));
c) Mobile Nodes Only: analyses if the approach take into account a network of mobile nodes only or it does also consider occasionally usage of static nodes, such as road-side units, to help in relaying messages;
d) Single-hop X Multi-hop: evaluates if a given approach consider routing through multiple nodes (multi-hop) or take message forwarding decisions from node to node (single-hop);
e) Test / Validation: provides information about the way an approach was tested;
f) Tools: reports the tools used to perform the tests.

Table 9.2: Summary Comparison of the Analysed Mission Dissemination in Mobile WSN Related Work.

Approaches	Node Density	Mobility Model	Mobile Nodes Only	Single-hop X Multi-hop	Test / Validation	Tools
Our Approach	Considered	Manhattan Mobility Model	Yes	Single-hop	Simulation	ShoX and GrubiX
GBMA	Not Considered	Random Waypoint Mobility Model	Yes	Multi-hop	Not mentioned	Not mentioned
MAQoS	Not Considered	Not mentioned	Yes	Multi-hop	Not mentioned	Not mentioned
Mobeyes	Considered	Real Track, Random Waypoint and Manhattan	Yes	Multi-hop	Simulation and Prototype	ns-2; custom environment
SHAFT	Considered	Manhattan Mobility Model	Yes	Single-hop	Simulation	OMNET++ and INET framework
VNUTS	Considered	Freeway Mobility Model	No	Multi-hop	Analytical Models	Mathematica

9.3 Alarm Handling in Cooperative Static and Mobile WSN

AWARE - HexDD

The AWARE project (ERMAN; HOESEL; HAVINGA, 2008) aims at integrating a sensor network of resource constrained ground sensor nodes with mobile sensors, carried on the ground by UGVs and in the air by UAVs. This project handles different concerns in relation to the cooperative use of mobile and static sensor nodes, from the interoperability issues to nodes' cooperation protocols. In (ERMAN; HAVINGA, 2010) a rendezvous-based data dissemination protocol to deliver critical alarm messages issued by the static sensor nodes to the mobile ones is presented. This approach divides the network of static sensors in a set of hexagonal cells, like a honeycomb, which is used in many other applications such as cellular phones. Hence, this data dissemination protocol is called Hexagonal Tiling-Based Data Dissemination (HexDD). The hexagonal structure of the network is used by the authors to specify regions of the network to where messages are forwarded to, as well as regions in which the static nodes maintain caches of the forwarded messages. Both alarm messages from the static sensor nodes and queries from the mobile ones are forwarded to the central hexagon of the network. Once a query reaches the central hexagon, the reply is sent in the reverse routing path. In the case that a query reaches a node that has a cached data that replies to the query, the information is sent in the reverse path without the need for the query proceed to the central hexagon.

The approach proposed in this thesis to deliver alarm messages from static sensor nodes to mobile ones, as presented in Chapter 5, share the same goals as the HexDD protocol presented in (ERMAN; HAVINGA, 2010). An important difference is that our pheromone-based approach mimicking the ants' behaviour is completely decentralized, as there is no preferable region of the network to where the messages should be forwarded. On the other hand, as their approach is based on rendezvous data dissemination that establishes regions where messages should be sent preferably, vulnerabilities such as traffic congestions and single point of failures have to be considered, while these are problems that do not affect our approach.

A Bio-Inspired Architecture for Division of Labour in SANETs (Bio-SANET)

In (LABELLA; DRESSLER, 2006) a problem very similar to the one presented in Section 1.2.3 is addressed, but the authors contextualize their work presenting the problem in a sensor/actuator network (SANET) scenario instead of in a wireless sensor network composed of static and mobile sensor nodes, as we do. In the context of SANETs, static sensors (sensors) provide information and send requesting messages to mobile robots (actuators) that respond these messages by moving to certain parts of the area in which the system is deployed. An example of application of such system is disaster relief, in which sensors send information that guide the movements of robots to assist victims. The addressed problem is divided in two parts: the message delivery from the static sensors to the mobile robots, and the task allocation.

The authors present a cross-layer approach in which the sensors and the robots are aware about the tasks that they are able to perform, which determines the destination of the messages sent through the network. The networking is based on the AntHocNet (CARO; DUCATELLE; GAMBARDELLA, 2004) and carried out by two phases, first

a route discovery is performed and then the routing. The routing discovery is a flood-like mechanism in which the `ROUTEDISCOVERY` messages are relayed among the sensor nodes until reach the destination. Once it reaches the destination, `ROUTEDISCOVERYRESPONSE` messages are sent back to the source node, collecting information about the link on the way, and informing the nodes in this way about the links of the previous visited nodes. Once the source node gets knowledge about one or more routes to the destination, it starts transmitting the data using the best routes.

The part of our work presented in Chapter 5 has many similarities with the work presented in (LABELLA; DRESSLER, 2006). Besides the mentioned similarity in relation to the context, the source inspiration based on the ant colony is also the same. In their model, sensor and robots are agents (physical agents), which makes our work very similar, as we model software agents (the nodeAgent) to control each node in our network. These two models are equivalent perspectives that provide the same practical result, as the nodes in our work can be seen as their physical agents. However, there are some important differences, as in our work the use of ants' pheromones and trails are not similar to their usage. In their approach the pheromones are used to update routing tables in the route discovery phase, which will provide trails representing the most promising routes to the destination. Moreover, they use a flood-based mechanism in the route discovery phase. On the other hand, in our approach trails correspond to pheromone marks left by the UAVs (our mobile sensor nodes) on the static sensor nodes, which are used to indicate their location so that messages can be forwarded in that direction and be delivered to them. Moreover, we avoid uncontrolled flooding by using the trail-search mechanism, and restricting flooding to the limits of the trail backbone.

Primate-inspired Scent Marking for Mobile and Static Sensors and RFID Tags (PRFID)

In (ZHANG; XIAO, 2009), an approach exploring the concept of scent marking used by primates for territorial demarcation is proposed to provide cooperation among mobile sensors, by using static sensors and RFID tags. In this proposal, mobile sensors are robots that move around in an area, and when they find events of interest in a given region, they leave RFID tags or sensors covering this region. If they need help to complete a given task, the information stored in the tags will indicate to other robots passing the region where help is required.

On one hand, this work is closely related to our pheromone-based alarm delivery because its concept of scent marking is very similar to our stigmergy inspired approach using pheromones to form the trails indicating the path followed by the mobile sensor nodes. The main difference from our work is that in their proposal the mobile robots deploy new nodes in the network, in an ad hoc fashion, as an attempt to disseminate the information that must be received by the other mobile nodes, in this case a help request. On the other hand, our approach assumes an already deployed network of static nodes, which may perform a number of sensing missions, with which the mobile ones cooperate by responding to alarms issued by the static nodes.

APEP

A proposal to solve the problem of message dissemination in a multi-level WSN composed by static and mobile sensors on the ground and UAV carried sensor nodes moving in the air is presented in (TAN; MUNRO, 2007). Their approach is based on the epidemic routing concept, but uses a probabilistic decision to forward or not incoming messages, so that a more efficient usage of the communication medium is achieved. As the nodes on the ground are also able to move, and considering the urban scenario addressed in their work, the nodes may have different number of neighbours during system runtime. Therefore, they propose the adaptation of the forwarding decision making process according to the nodes' neighbourhood and from this aspect come the name of the proposal: APEP (Adaptive Probabilistic Epidemic Protocol).

An aspect of APEP that is similar to our work is the need to deliver messages from the ground sensors to the UAVs. As the positions of the UAVs are unknown, they try to address the problem by a flooding-based method, whose overhead is reduced by the probabilistic decision making process and by the knowledge about the neighbourhood of a node on the ground. Our approach is different because it tries to provide information about the UAVs' movement to the ground nodes, so that the ground nodes can forward messages in the direction of the most suitable UAVs, thus reducing the drawbacks typical of flooding-based mechanisms. Even the blind trail-search that has to be performed in the cases in which alarms are issued by nodes that do not have the pheromone information indicating the direction of a UAV, the costs are considerably lower than those accounted for in a conventional flooding, as presented in Section 5.5.

Surveillance and Tracking System with Collaboration of Robots, Sensor Nodes, and RFID (STRFID)

In (XIAO; ZHANG, 2009), a divide and conquer solution for surveillance of large areas is proposed in which static and mobile sensors provide coverage in an area. Static sensors are deployed to detect and keep track of events of interest in selected regions, such as the boundaries of the area, while mobile sensors move around these regions collecting data from the static nodes. The coverage problem is also studied in (BRASS, 2007), in which the capability of area coverage by regularly and randomly deployed sensors is compared.

The main difference to our approach is that they consider a favourable distribution of the nodes to carry out the cooperation among different types of nodes, analyzing which node distributions that increase the area coverage. This aspect is out of the scope of our approach which considers the random deployment of the static sensor nodes and random mobility for the mobile sensor nodes.

Whisper

Whisper (Wireless High Speed Routing) (OLIVEIRA et al., 2010) presents a proposal to address the problem of routing data from static sensor nodes to a mobile sink node. This problem can be mapped to the scenario presented in Section 1.2.3, considering the UAVs as a type of mobile sink. Whisper proposes a solution focusing on high speed mobile sinks. The sink nodes send queries that are flooded on the network of static sensor nodes, together with the information about the trajectory of its movement. The replies of these queries are routed back to the mobile sink node from the query replying nodes by calculating a future position of the mobile sink in relation to the one from which it sent the query. The calculation of the future position is done by

each intermediary node in each hop, from the replying node until the message reaches the mobile sink. To perform this calculation the nodes use the information about the sink's trajectory. Three different variations are presented by the authors, which varies with respect to the selection of the mobile sink's future position that is calculated. The first variation (Whisper Follow) basically computes the current position of the mobile sink in each hop and forwards the reply message to a node that is closer to this position. The second variation (Whisper Intercept) calculates the first point of interception between the trajectories of the mobile sink and the reply message. The third variation (Whisper Shortest) selects as destination point the one that provides the shortest path linking the sending node and the trajectory of the mobile sink node.

Despite the goal to deliver messages from static sensor nodes to mobile nodes Whisper has an important difference in relation to our approach: its mobile sink sends and floods the queries among the sensor nodes in the network. This difference is also valid for other works that focus on the use of mobile sinks. On the other hand, in our case, the queries (the missions in our case) are supposed to be already known by the static sensor nodes, and they instead have to reach the mobile sensor nodes (UAVs) with alarm messages upon the detection of an event of interest. Considering the scenario presented in Section 1.2.3, it is possible to imagine that the UAVs could periodically flood the network with queries to ask for data about detected threats, but this approach would be highly inefficient from the energy savings perspective.

The techniques used for mobile sinks are related to our study, and Whisper presents interesting features that could be further explored to enhance our approach or to provide a hybrid approach. For instance, it is possible to use their technique to relax the feasibility condition for the minimum beacon period required in our approach to maintain the consistency of the pheromone trails, as analysed in Section 5.4. This would be possible by using their idea for the calculation of the destination of the query response messages based on the sink trajectories to cover gaps that appear in the pheromone trail if the beacons are not sent within the minimum beacon period. A hybrid approach could be possible by joining our trail-search to one of the Whisper variations. During the trail-search, once an alarmAgent reach a node that have received a beacon from a UAV, it could follow the Whisper algorithm to reach the UAV, as the node would then have the UAV trajectory information received in the beacon sent by the UAV.

Other Related Works within the Research Area of Mobile Sinks

Other related approaches (DEMIRBAS; SOYSAL; TOSUN, 2007) (HWANG; IN; EOM, 2006) (JAIN et al., 2006) (OCHIAI et al., 2010) handle the problem of mobile sinks in WSNs, which can also be compared to ours. Some of these proposals consider that the mobile sinks may decide about their movement in order to facilitate the message delivery by static sensor nodes and thus optimize the energy usage in the network as a whole (DEMIRBAS; SOYSAL; TOSUN, 2007) (HWANG; IN; EOM, 2006). Other proposals assume that the movement of the sinks is at least predictable, as in (JAIN et al., 2006), which is also the case of Whisper (OLIVEIRA et al., 2010) as presented above. The assumptions about mobility presented in these works differ from ours, as there is no predictability in the movement pattern of the UAVs in our work. Moreover,

the UAVs cannot present preference to move towards the direction of a given group of nodes, before the occurrence of an alarm, as there is no prediction or other indication that they would be needed in a specific location.

Also considering mobile sinks, in (OCHIAI et al., 2010) the concept of delay tolerant networks (DTN) is explored. The authors report experiments with the deployment of mobile nodes to collect information from static sensor nodes to be used in agricultural applications. To drive the movement of data from the sensor nodes towards a central processing node via the mobile sink, the authors use a potential-based routing mechanism that sets higher potential to data providers, intermediary potential to the mobile sinks and a lower one to the central node. Their potential-based routing is to some extent comparable to our pheromone-based one, but, while the pheromone-based mechanism addresses the dynamicity imposed by the movement of the UAVs, their potential-based mechanism is statically configured before the system runtime.

Summary Comparison

Table 9.3 complements the analysis performed of the main related work in this section by a comparison tanking into account the following criteria:

a) Networking: evaluates the networking mechanisms used by each approach;
b) Nodes' Interaction: evaluates if an approach considers direct communication between static and mobile sensor nodes, or uses a gateway node to provide this communication;
c) Mobile Nodes Mobility Models: reports the mobility models used by each approach;
d) Static Nodes Distribution: informs the distribution used to spread the static nodes in the test environment;
e) Test / Validation: provides information about the way an approach was tested;
f) Tools: reports the tools used to perform the tests.

Table 9.3: Summary Comparison of the Analysed Cooperation among Static and Mobile Wireless Sensor Nodes Related Work.

Approaches	Networking	Nodes' Interaction	Mobile Nodes Mobility Model	Static Nodes Distribution	Test / Validation	Tools
Our Approach	Geographic Routing and Controlled Flooding	Direct Communication	Random Walk and Predefined Waypoints	Random and Regular Grid	Simulation and Partial Prototype	ShoX, GrubiX, SunSPOTs and AFME
APED	Probabilistic Flooding	Direct Communication	Obstacle Mobility Model and Circular Trajectory	Arbitrary Predefined Positions	Simulation	Not mentioned
AWARE (HexDD)	Rendevouz Based	Communication Via Gateways	Linear Mobility Model	Random	Simulation	ns-2
Bio-SANET	Flooding and Shortest Path Routing	Direct Communication	Predefined Waypoints	Regular Grid	Simulation	OMNeT++ and ODE
PRFID	Opportunisitc Data Delivery	Direct Communication	Linear Mobility Model by Segments	Random	Simulation	Custom environment
STRFID	Opportunisitc Data Delivery	Direct Communication	Predefined Trajectories: circular and arbitrary closed shape curve	Optimal Area / Perimeter Coverage	Simulation	Custom environment
Whisper	Flooding and Rendevouz Based	Direct Communication	Linear Mobility Model	Non-Regular Grid	Simulation	ns-2

10 CONCLUSION

The structure and presentation flow used in this thesis is based on the three scenarios described in Section 1.2. Following the same organization, the conclusions for solutions proposed for the problems of each of these three scenarios are presented in the same way, followed by a discussion about future work and final remarks.

10.1 Sensing Mission Dissemination and Allocation in Static WSN

The scenario described in Section 1.2.1 presented the context in which static sensor nodes cooperate to perform sensing missions. From this overall scenario, the problems of mission dissemination and allocation were selected to be handled in this thesis work.

The mission dissemination is addressed by a geo-aware strategy to drive the mission towards its area of interest (the mission area) and, when arriving there, to spread the mission among the sensor nodes in an energy use concerned way, by avoiding unnecessary and redundant transmissions, i.e. reducing communication to help in energy savings. The mission allocation is addressed by an autonomous decision procedure performed in each node, which explores the concept of locality, considering the conditions of neighbouring nodes. This decentralized decision procedure does not require any additional exchange of information among the nodes, besides the information that was already carried by the missionAgents during the mission dissemination phase. Borderline adverse situations expressed by regions consisting of concentration of nodes with similar conditions are handled by the introduction of an additional bio-inspired agent, the beeAgent, so that these situations could be bypassed.

Simulations were performed in which the proposal was tested under general conditions, with a uniform random distribution of the nodes placement, and in another more specific distribution modelling the existence of node concentrations. In both cases the approach was tested with the additional beeAgent and without it. Experiments indicate affordable communication overhead and good mission allocation results in relation to the mission requirements, regarding both the number of nodes selected to engage in the mission and their quality to perform the mission. The results achieved were similar for the random distributions with and without the beeAgent, which shows that this additional agent does only interfere in the cases in which regions of node concentrations are identified. In such cases, better results were achieved with the beeAgent. Compared to a flooding reference solution, the proposed approach achieved better results, having the cost in terms of communication among the sensor nodes 30% to 40% lower than the a flooding based solution for the cases without regions of concentrations. For the cases in which regions of concentration were present, the cost results were around 50% lower compared to the flooding based solution. Compared to the optimum reference, the proposed approach requires in average twice as much communication. Despite this significant difference, this result is fairly good considering the nature of the mission dissemination, which is to inform all the sensor nodes about a given mission. The optimum uses the privileged information about the mission and the

nodes placement to select those nodes that will proceed forwarding the mission, which is in practice impossible to be implemented.

The simulations also explored an increase in the communication range. Our experiments indicate that the increased range represents a small contribution in enhancing the results, especially for the case with concentration of nodes (requiring approximately 6% less communication in average). These results provide evidence that, for the same average node density, a decision to increase the communication range has to carefully consider its benefits and associated drawbacks. This consideration is motivated especially because when increasing the communication range, more energy is spent to send data, thus negatively impacting the energy consumption.

10.2 Sensing Mission Dissemination in Mobile WSN

For the scenario presented in Section 1.2.2, the problem of mission dissemination in a sensor network composed of mobile sensor nodes is analysed. In this scenario the problem was not divided in two as it was the case for the network composed of static sensor nodes. The reason for this is because once a node gets the mission in this scenario, it is assumed that the node can be selected to perform it, i.e. the mission is allocated to the node.

Taking into account the dependence of the missions in relation to their respective mission areas, a study about the use of geographical context information was performed, in which the goal was to keep a mission as long time as possible hosted by nodes located inside its respective mission area. To perform this study, the missions were carried out by mobile agents that encapsulate both the mission and the decision mechanism telling how to migrate from node to node. Three different levels of intelligence were established considering different degrees of detail in the geographical information used for the decision to migrate to another node or to stay in the current hosting node. These proposed levels were compared among themselves and to a reference solution that uses a random choice between if to migrate or not.

The results provided interesting insights for the behaviour of the agents according to their level of intelligence. One important observation is that agents with lower levels of intelligence are more susceptible to differences in node density than agents with higher levels of intelligence. For the former, the performance enhances significantly in an environment having a high density of nodes, while the difference in the performance for the latter is much less remarkable. The agents with lower levels of intelligence almost double the percentage of time that they manage to keep inside the MA in scenarios with high density of nodes, while the more intelligent ones enhance their results with approximately 50%. This can be seen as good or bad depending on the perspective of the analysis. Considering environments that always present high density of nodes, in spite of the best results achieved by the agents with higher levels of intelligence also in these cases, the good results achieved by the less intelligent agents may discourage the usage of the more intelligent ones. On the other hand considering a more general case, in which the environments vary in the density of nodes, definitely the use of the more intelligent agents present clear advantages in relation to the less intelligent ones, as they are able to keep a more uniform and stable pattern of performance in both high and low

density cases. Moreover, it is also important to highlight that this remarkable enhancement in the results achieved by the agents with lower levels of intelligence is associated with an equivalent increase in the cost due to communication cost.

Considering the communication usage, the Route Aware approach, representing the highest level of agent intelligence, achieved the best savings. Following the same pattern, the direct path approach is better than the Random Reference approach used for comparison, but worse than the Destination Based one. This is interesting finding that can be used to evaluate the pros and cons in using more or less of agent intelligence depending on the kind of node that is being considered. For instance, for nodes that have severe energy resource restrictions and cannot use the Route Aware approach, a better alternative would be to use the Destination Based one instead of the Direct Path, even considering that the Destination Based does not manage to keep the mission inside the mission area as much as the Direct Path. However, the Destination Based approach provides an alternative that saves energy, as it needs to communicate less than the Direct Path one. On the other hand, if the concern about energy is not a special issue, which is the case to nodes with large amount of available energy resources, if the Route Aware approach is not possible to be used, the Direct Path is a good alternative.

10.3 Alarm Handling in Cooperative Static and Mobile WSN

For the alarm delivery and assignment problem related to the scenario presented in Section 1.2.3, a decentralized solution based on the behaviour of ants was proposed. The goal was to provide a solution that could fulfil the needs for communication among static and mobile sensor nodes, while keeping the communication overhead as low as possible.

Particularly, the considered mobile sensor nodes are aerial robots, Unmanned Aerial Vehicles (UAVs), which may carry different types of sensors. Following a description of a system in which the cooperation among static and mobile sensor nodes is performed by means of alarms issued by the static sensors to be delivered to the UAVs, by means of mobile software agents (the alarmAgents), the problem was analysed in two steps. The first step considered all UAVs carrying the same type of sensor, thus the goal was to find any UAV upon the occurrence of an alarm. The second step considered UAVs carrying sensors with different capabilities that provide different results depending on the type of threat.

For the first step a solution based on the pheromone trail construction and following behaviour presented by ants was proposed. According to this proposal, once an alarm was issued, an alarmAgent will search for a trail and once it find one it will follow it until it can deliver the alarm to the respective UAV. This solution was then enhanced by means of differentiating the pheromones left by the UAVs to form their trails. This enhancement considered that according to the type of the sensor carried by the UAV, it sends pheromones with different flavours, which attract alarmAgents responsible to deliver alarms informing about different types of threats. Further enhancements were performed in this approach, in which alternative variations for the dissemination of the pheromones of the different UAVs were presented. Additionally, a feasibility analysis of the basic technique was performed.

Simulations evaluate the performance of the proposed approach and its variations, according to the defined metrics of interest. Comparisons between the results achieved by the proposed approach and the reference solutions were presented. Encouraging results were achieved, providing evidence that the proposed method to carry out the cooperation among static and mobile sensors presented in this work is efficient to address the formulated problem. In average the proposed pheromone based approach requires 10 times less communication among the static sensor nodes than the flooding reference. For the variations considering mobile sensor nodes with different sensing capabilities, this difference is attenuated but is still big (around 3 times in the worst case comparing heuristic-Pf-hbt to flooding). In relation to the results assessed for the utility in engaging a given UAV to handle a given alarm, the enhanced variations of heuristic-Pf performed better, around 10% comparing heuristic-Pf-hbt to heuristic-Pf. However, as already mentioned, heuristic-Pf-hbt has a higher cost in terms of communication, thus a trade-off between the cost and utility metric is stated and has to be considered in each specific situation of utilization of the different variations of the basic heuristic.

The analysis of the achieved results also provided other interesting findings, such as the relation between the movement pattern used by the UAVs and the overhead in terms of communication among the static sensor nodes due to the trail-search and trail-follow mechanisms. It was assessed that longer trails increase the overhead due to the trail-follow and diminish the overhead due to the trail-search. Conversely, shorter trails diminish the overhead due to the trail-follow and increase the overhead due to the trail-search.

10.4 Future Work

The study developed in this thesis provides a background and base for a number of future works that can extend and enhance the proposed solutions for the addressed problems. From the perspective of a complete surveillance system that is able to cover the three specific analysed scenarios, as presented in Section 1.2, several possible future works can be developed to integrate the proposals of our thesis. Practical issues related to testbed deployments and enhancements in the simulations and in the GrubiX simulator can also be named in the future work directions.

Observing the possibilities provided by this thesis work from these different perspectives, firstly specific future works related to each of these three core parts are discussed. This is followed by a discussion related to enhancements in the simulations and in the simulation tool, and to the deployment of testbeds, which are valid for all tree core parts of this work. Finally, a discussion about future works related to the conception of an integrated system covering the whole motivation scenario is presented.

10.4.1 Mission Dissemination and Allocation in Static WSN

For the solution approach presented in Chapter 3, considering the mission dissemination phase, our approach assumed that once a sensor node broadcasts an agent, it is received in each sensor node and processed accordingly. In the case in which the agent should not remain in a sensor node, it is just discarded. An alternative for this way to transmit the agents would be to let the sensor nodes to keep and maintain a table with

information about its neighbours and before the agents are sent, they would consult this table and specifically decide to which node they should be transmitted. The positive effect of this method would be to diminish the overhead caused by processing the agent decision once it arrives at a node. The negative is that, this would require the maintenance of the table with information about the neighbours, which can lead to an increase in the overhead due to communication to send and receive beacons to update this table.

In relation to the decision procedure needed to decide if or not to engage a sensor node to take part in performing a mission, alternative methods could be used to evaluate its suitability to a given mission in relation to its neighbours. In our approach a weighted probability calculation was used, but other methods such as the Gaussian process used in (KHO; ROGERS; JENNINGS, 2009) could also be tested. Variations of our method could also be tested, for instance by varying the value used in the comparison to the one obtained in (3.1).

Adaptation in the mission allocation is also an issue that can be addressed to enhance our proposal. The missionAgents can remain in the sensor nodes eligible to perform a mission, even after it is decided that these nodes will not be allocated for that mission, and perform a periodic evaluation of the engagement of the node or upon the occurrence of a given event. The nodes could keep old missionAgents of missions that they are not engaged as long as they have available memory space to store them and the new incomer missionAgents.

10.4.2 Mission Dissemination in Mobile WSN

Our study about the mission dissemination in mobile wireless sensor networks presented in Chapter 4 can be enriched by other proposals for how to do the intelligent decision about the missionAgents' migration. In our study three agent approaches with different intelligence levels was considered and compared both among them and against a reference solution that offered a random decision approach. Other types of decision algorithms can be proposed considering other types of information, such as traffic congestions and speed of the nodes for the analysed VSN case study.

The environment model can be enhanced to make available other information that can be used in the decision algorithm. In our experiments a simple model of a city divided in regular blocks was used, but it is possible to extend these experiments with models that use more realistic scenarios, or even real route maps of cities.

10.4.3 Alarm Handling in Cooperative Static and Mobile WSN

The proposal presented in Chapter 5 is definitely the most important part of our work, which was the most explored and which is the one that can be used as basis for future research work.

During the trail-search, the solution described in Section 5.4 establishes that once an alarmAgent finds a trail it starts the trail-follow process in the heuristic-P. Considering UAVs equipped with different types of sensors, instead of the heuristic-P, the heuristic-Pf and its variations are used. According to the heuristic-Pf, the flavour of the pheromones is considered by the alarmAgent when it finds a trail, which means that it does not only take the first trail that it finds, but it analyse it before switching to the trail-follow mechanism. However, in this approach the alarmAgent considers that it is

enough that the trail presents a pheromone of an UAV that has a sensor capable to handle the threat that its alarm announces, but it does not analyse if it is the best sensor for that alarm. An alternative solution that can be investigated is to make the alarmAgent continue the trail-search until it finds the trail of the UAV equipped with the best sensor, assess the associated costs and analyse them in relation to the acquired benefits.

The overhead due to the trail-follow mechanism can be diminished if instead of using a controlled flooding within the trail limits, the alarmAgent be forwarded by just one node at a time, hop by hop on the way towards to the UAV, following the strength of the pheromone marks. Considering the node density used in our experiments, this approach would result in about 5 times fewer forwarded messages in average in relation to the results presented in Chapter 5. However, we opted for the controlled flooding due to the fact that we used a network of static sensor nodes in the limit of the connectivity, i.e. with a number of nodes that is enough to keep the network connected, but close to becoming disconnected. As analysed in Section 8.3.3, gaps in the connectivity are really a problem that may affect the forwarding process. Hence, further studies in which networks with higher density of nodes applying unicast forwarding instead of a flooding inside the pheromone trails have to be considered, as well as proposals to address the dependability problems related to the low node density, as discussed in Section 8.3.3.

As analysed in Sections 5.5.3 and 5.5.4, the mobility model adopted by the UAVs has a significant influence in the costs due to the trail-search and trail-follow mechanism used in the proposed pheromone-based approach. Further studies about the influence of different movement patterns are desirable, especially relating these cost effects to other factors associated to other non-functional requirements, such as secrecy.

The complete model of the UAV carried mobile sensor nodes described in Section 5.2, Figure 5.2, was not implemented for the experiments performed in this work. The implementation of this complete model will allow the assessment of the benefits of the interaction among the mobile sensors to exchange alarms. This may represent an important analysis especially for higher numbers of UAVs and threats, in which the UAVs maintain long alarm pending queues, and as being more numerous in the scenario, the UAVs would meet more often creating possibilities to interact and exchange or handover alarms. Still related to this topic, the negotiation could be done by making the alarmAgents act somehow similarly as the missionAgents in the solution described in Chapter 4, in which they decide by themselves to migrate to a meeting node. Likewise, the alarmAgents could be stored by the UAVs, and upon a meeting between UAVs, the alarmAgents would decide by themselves to migrate to the meeting UAV, or stay in the current one.

In our experiments we considered static threats. However, it is most likely that a surveillance system that uses such a solution also intends to monitor mobile threats. Thus a tracking mechanism has to be integrated in the proposed solution in which once a sensor node detects a mobile threat and sends the alarmAgent, it also triggers the tracking of this mobile threat. Then, when the UAV arrives to the location where the alarm was issued it can query the sensor node that issued the alarm and this node can inform about the current position of the threat, so that the UAV can reach the mobile threat. Another improvement to make the experiments more realistic would be to

introduce an overhead due to the alarm handling in terms of time spent by the UAV to handle a given threat. The current implementation uses a simplification considering that the UAVs complete handling a given threat by arriving to the location where the respective alarm was issued.

10.4.4 Simulation

The performed simulations aimed to assess the efficiency in relation to the specified requirements in each of the studied scenarios and the related overhead of the proposed solutions. However, as mentioned in Section 1.7 the experiment design space is very huge, with a large number of possible variations due to several involved variable factors. A natural continuation of this work is to enrich the set of simulations to test variations in other factors of importance that were not considered in this work and their further statistical analysis trying to discover relations among the factors that were identified in the experiments performed so far.

GrubiX is an evolving simulation tool which provided the necessary support to test the solution approaches proposed in this work. However, there are a number of new features that would be highly desirable to be added to extend its functionalities so that it can provide even better assistance in running experiments such as those presented in our work. One of these possible extensions is to give support for agent-oriented programming. This feature could be added by providing an interface with an agent-oriented programming environment to run on top of the application layer provided by the simulator framework. A possibility in this direction that is being studied is to integrate support for Jason (BORDINI; HUBNER; WOOLDRIDGE, 2007) to write the code for software agents. Despite GrubiX existing support for energy and interference models, these two features are not very well explored neither developed yet. Efforts to implement realistic energy models for the sensor nodes and interference models for the environment represent an important direction for the future works to provide more realistic simulations. Energy modelling would make it possible to perform simulations aimed to test the system lifetime as well as to provide results for the overhead in terms of energy consumption instead of the number of packets sent due to our proposed algorithms. The use of interference models would make it possible to evaluate how interference in the communication among sensor nodes could affect the results of our proposed solutions.

The visualisation tool also needs to be enhanced to provide a three dimensional perspective of the simulated environment. Another enhancement that can be done is to provide different visual icons to illustrate the nodes, according to their different types, besides of the existing difference in the colours that can be attributed to them. The possibility to load maps for the environment would also be a desirable feature. Moreover, in relation to this feature, it could even extend the functionality of the simulation itself and not only for the visualisation tool, by making it possible to load maps with embedded semantics. Contour maps could provide information about altitude variations that may create hinders to the communication or sensing that can interact with the communication and application models used in the simulations.

10.4.5 Physical Testbed

A natural continuation of the work presented in this thesis is to go from simulation to the implementation of real physical wireless sensor networks equipped with real sensor nodes and to deploy them in reasonably large testbed, one for each solution presented in Chapters 3, 4 and 5. This should be followed by a larger testbed integrating the three parts of the work and creating a base for the possible future works discussed in the previous section.

Despite its small scale, the demonstrator presented in Chapter 7 is the beginning of such future work direction towards the deployment of testbeds that implement our proposed solutions. The intention in deploying the presented demonstrator had no ambition to cover the different aspects that a medium or large scale testbed can address, but it can help to study system integration properties if further developed. The next step in the development of this demonstrator to enable it for outdoor deployments is to make a connection between the sensor node carried by the quadcopter and the navigation board of the mobile platform. This connection would allow the transmission of GPS coordinates from the carried sensor node to the navigation board upon the reception of an alarm. The navigation board would then take this coordinates and set them as the next destination of the quadcopter.

As mentioned in Section 1.7, and besides the small scale demonstrator presented in Chapter 7, the methodology for experimentation of the proposed solutions was based on simulations due to practical reasons and lower costs compared to physical testbeds and experiments. The continuation of this work in the direction of the deployment of our solutions in real sensor node platforms it is of great value. This to assess how effective they are in the real world, and how they can be enhanced to cover gaps that may hinder their efficiency when running on real deployed systems.

10.4.6 Fully Integrated Cooperative Static and Mobile WSN

As presented in Section 1.2, the usefulness of a surveillance system is highly enhanced if it provides the possibility to use different kinds of sensor nodes according to the user needs and the operation conditions. In relation to the specific difference between the sensor nodes addressed in our work, i.e. the mobility capabilities, this aspect affects a number of system features, such as coverage and secrecy. As already discussed in Chapter 1, the use of cheap static sensors nodes allow the massive usage needed to cover large areas and to detect events of interest with sufficient probability. On the other hand, the use of mobile sensor nodes allow the employment of more sophisticated sensors that can move around in an area of interest according to the needs, thus providing a flexible way to cover a large area with a more sophisticated sensor. For the secrecy, the use of mobile sensors in urban areas is appropriate as they are hardly identified among the other vehicles. On the other hand, small static sensor nodes are appropriate to deployments in environments where they can be camouflaged, such as under foliage.

There are a number of possibilities to combine the different types of sensor nodes in different deployments; in fact this feature enhances much the functionalities of a surveillance system. However, some issues have to be addressed to make this approach feasible, in which the first to be mentioned is the integration of the different types of

sensor nodes and their combination, in a unified network. This is an issue that would be of great value to be addressed in futures works. Another important aspect relates to the autonomy of the system needed to choose the best type of sensors to accomplish a given mission, based on its requirements. This feature frees the user of the details of how to setup the system and thus it is highly desirable from the user friendliness perspective. Future works in this area are also expected to address the high level user interface, and could include the proposal of graphical interfaces and high level mission specification languages.

10.5 Final Remarks

This thesis work provides contributions to the area of surveillance systems based on wireless sensor networks technology employing static and mobile sensor nodes. Solutions for sensing mission dissemination and allocation, and cooperation among the sensor nodes were presented in which all of them considered the concern about energy consumption, by making efforts to diminish the communication among the sensor nodes.

Besides the described contributions, we believe that this thesis provides the foundation for many future works that can enhance and extend what was proposed, aiming to provide more efficient solutions for the use of wireless sensor networks in surveillance systems.

REFERENCES

ADNER, R.; LEVINTHAL, D. Demand Heterogeneity and Technology Evolution: Implications for Product and Process Innovation. **Management Science**. [Sl]:INFORMS, v.47, n.5, p.611-628, 2001.

AIELLO, F.; FORTINO, G.; GRAVINA R.; GUERRIERI, A. A Java-Based Agent Platform for Programming Wireless Sensor Networks. **The Computer Journal.** Oxford, UK: Oxford Journals, v.54, i.3, p.439-454, 2011.

AKYILDIZ, I.; KASIMOGLU, I. Wireless sensor and actor networks: research challenges. **Ad Hoc Networks**. [S.l.]:Elsevier, v.2, i.4, p.351-367, 2004.

AKYILDIZ, I. F.; SU, W.; SANKARASUBRAMANIAM, Y.; CAYIRCI, E. A survey on sensor networks. **IEEE Communications Magazine.** Washington, USA: IEEE Computer Society, v.40, i.8, p.102-114, 2002.

ALEJO, D.; CONDE, R.; COBANO, J.A.; OLLERO. A. Multi-UAV collision avoidance with separation assurance under uncertainties. In IEEE INTERNATIONAL CONFERENCE ON MECHATRONICS, 2009, Malaga, Spain. **Proceedings…** Washington: IEEE Computer Society, 2009. p.1-6.

ALEMDAR, H.; ERSOY, C. Wireless sensor networks for healthcare: A survey. In **Computer Networks.** New York, NY, USA: Elsevier, v.54, i.15, p. 2688–2710, 2010.

ANTHONY M.; BARTLETT, P. L. **Neural Network Learning: Theoretical Foundations**. Cambridge, UK: Cambridge University Press, 2009.

ANTONIOU, P.; PITSILLIDES, A.; BLACKWELL T.; ENGELBRECHT, A. Employing the flocking behavior of birds for controlling congestion in autonomous decentralized networks. In IEEE Congress on Evolutionary Computation, 2009, Trondheim, Norway. **Proceedings…** Washington: IEEE Computer Society, 2009. p. 1753-1761.

ASHBY, W. R. Principles of the Self-Organizing System. In VON FOERSTER, H.; ZOPF, G. W. (Eds.). **Principles of Self-Organization.** London, UK:Pergamon Press, 1962, p.255-278.

AVIZIENIS, A.; LAPRIE J.; RANDELL, B. Fundamental Concepts of Computer System Dependability. In IARP/IEEE-RAS WorKSHOP ON ROBOT DEPENDABILITY: TECHNOLOGICAL CHALLENGE OF DEPENDABLE ROBOTS IN HUMAN ENVIRONMENTS, 2001, Seul, South Korea. **Proceedings…** Washington, USA: IEEE Computer Society, 2001. p.1-16,

BAI F.; HELMY, A. A Survey of Mobility Models in Wireless Adhoc Networks. **Wireless Ad hoc and Sensor Networks.** [S.l.]:Kluwer Academic Publishers, v.2, i.5, p.1-30, 2006.

BAI, F.; SADAGOPAN N.; HELMY, A. IMPORTANT: a framework to systematically analyze the Impact of Mobility on Performance of Routing Protocols for Adhoc Networks. In IEEE INFORMATION COMMUNICATIONS CONFERENCE, 2003, San Francisco, USA, **Proceedings…**[S.l.:S.n.], 2003. p.825-835.

BARDELABEN, J.A. Multimedia sensor networks for ISR applications. In CONFERENCE RECORD OF THE THIRTY-SEVENTH ASILOMAR CONFERENCE ON SIGNALS, SYSTEMS AND COMPUTERS, 2003, v.2. **Proceedings…**[S.l.:S.n.], 2003. p.2009-2012.

BEARD, R.W.; MCLAIN, T.W.; NELSON, D.B.; KINGSTON D.; JOHANSON, D. Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs. **Proceedings of the IEEE**. Washington, USA: IEEE Computer Society, v.94, i.7, p.1306-1324, 2006.

BELLIFEMINE, F.; CAIRE, G.; POGGI A.; RIMASSA, G. Jade a white paper. Telecom Italia **EXP Mag.** [S.l.:S.n.], v.3, n.3. p.6–19, 2003.

BETTSTETTER, C. On the Minimum Node Degree and Connectivity of a Wireless Multihop Network. In 3rd ACM INTERNATIONAL SYMPOSIUM ON MOBILE AD HOC NETWORKING & COMPUTING, 2002, Lausanne, Switzerland. **Proceedings…**[S.l.:S.n.], 2002. p.80–91.

BONABEAU, E.; DORIGO, M.; THERAULAZ, G. **Swarm Intelligence: From Natural to Artificial Systems**. Santa Fe Institute Studies in the Sciences of Complexity, NY: Oxford University Press, 1999.

BONNET, P.; GEHRKE J.E.; SESHADRI, P. Towards sensor database systems. In 2nd INTERNATIONAL CONFERENCE ON MOBILE DATA MANAGEMENT, 2001, Hong Kong, China. **Proceedings…**[S.l.:S.n.], 2001, p.3-14.

BOONMA, P.; SUZUKI, J. BiSNET: A biologically-inspired middleware architecture for self-managing wireless sensor networks. **Computer Networks**. [S.l.]: Elsevier, v.51, i.16, p.4599–4616, 2007.

BORDINI, R. H.; HUBNER, J. F.; WOOLDRIDGE, M. **Programming Multi-Agent Systems in AgentSpeak Using Jason**. Chichester, UK: John Wiley & Sons Ltd, 2007.

BOTTS. M. **Sensor Model Language (SensorML) for In-situ and Remote Sensors**, DIPR Version 0.7, Technical report, Open GIS Consortium (OGC), [S.l.:S.n.], 2002.

BOX, G. E. P.; HUNTER, J. S.; HUNTER, W. G. **Statistics for experimenters Design, innovation and discovery**. 2.ed. New Jersey: John Wiley & Sons Ltd, 2005.

BRASS, P. Bounds on coverage and target detection capabilities for models of networks of mobile sensors. **ACM Transactions on Sensor Networks**. New York: ACM, v.3, i.2, p.9-es, 2007.

BRATMAN, M. E. **Intention, Plans, and Practical Reason**. Cambridge, USA: University of Chicago Press, 1987.

BRUCE, K. B. **Foundations of Object-Oriented Programming Languages: Types and Semantics**. Massachusetts, USA: The MIT Press, 2002.

CAPUTO, D.; GRIMACCIA, F.; MUSSETTA M.; ZICH. R. E. Genetical Swarm Optimization of Multihop Routes in Wireless Sensor Networks. **Applied Computational Intelligence and Soft Computing**. v. 2010, Article ID 523943, 14 p., 2010.

CARO G.; DORIGO, M. AntNet: Distributed Stigmergetic Control for Communications Networks. **Journal of Artificial Intelligence Research**. [S.l.]: AI Access Foundation, v.9, i.1, p.317-365, 1998.

CARO, G.; DUCATELLE F.; GAMBARDELLA, L.M. AntHocNet: an Ant-Based Hybrid Routing Algorithm for Mobile Ad Hoc Networks. In 8th INTERNATIONAL CONFERENCE ON PARALLEL PROBLEM SOLVING FROM NATURE, 2004, Birmingham, UK. **Proceedings…**[S.l.]: Springer, p.461-470, 2004.

CASTRO, L.N.; TIMMIS, J. **Artificial Immune Systems: A New Computational Intelligence Approach**. London, UK: Springer, 2002.

CASTRO, L. N.; ZUBEN, F. J. From Biologically Inspired Computing to Natural Computing. In CASTRO, L. N. ZUBEN, F. J. (Eds). **Recent developments in biologically inspired computing**. Hershey, USA: Idea Group Publishing, 2005, p.1-7.

ÇELIK, F.; ZENGIN, A.; TUNCEL, S. A survey on swarm intelligence based routing protocols in wireless sensor networks. **International Journal of the Physical Sciences**. [S.l.]: Academic Journals, v.5, i.14, p.2118-2126, 2010.

CHEN, J.; KHER S.; SOMANI, A. Distributed fault detection of wireless sensor networks. In Workshop on Dependability Issues in Wireless Ad hoc Networks and

Sensor Networks, 2006, Los Angeles, CA, USA. **Proceedings…**[S.l.:S.n.], 2006, p.65-72.

CHEN, D.; VARSHNEY, P.K. QoS Support in Wireless Sensor Networks: A Survey. In International Conference on Wireless Networks, 2004, Las Vegas, Nevada, USA, 2004. **Proceedings…**[S.l.:S.n.], p.227-233.

CHESS, D.; GROSOF B.; HARRISON. C. **Itinerant agents for mobile computing**. Technical Report RC 20010, IBM Research, 1995.

CHONG C-Y.; KUMAR, S.P. Sensor Networks: Evolution, Opportunities, and Challenges. **Proceedings of the IEEE**. Washington, USA: IEEE Computer Society, v.91, i.8, p.1247-1256, 2003.

CHRISTIN, D.; REINHARDT, A.; MOGRE P.S.; STEINMETZ, R. Wireless Sensor Networks and the Internet of Things: Selected Challenges. In 8th GI/ITG KUVS FACHGESPRÄCH "DRAHTLOSE SENSORNETZE", 2009, Hamburg, Germany. **Proceedings…**[S.l.:S.n.], 2009. p. 31-33.

COHEN P. R.; LEVESQUE, H. Intention is choice with commitment. **Artificial Intelligence**. Essex, UK: Elsevier Science Publishers Ltd., v.42, i.2-3, p.213-261, 1990.

COLTIN, B.; VELOSO, M. Mobile robot task allocation in hybrid wireless sensor networks. In IEEE/RSJ International Conference on Intelligent Robots and Systems, 2010, Taipei, Taiwan. **Proceedings…**[S.l.:S.n.], 2010, p.2932–2937.

CONTI, M.; CROWCROFT, J.; GIORDANO, S.; HUI, P.; NGUYEN, H.A.; Passarella, A. Routing Issues in Opportunistic Networks. In GARBINATO, B.; MIRANDA, H.; RODRIGUES, L. (Eds.). **Middleware for Network Eccentric and Mobile Applications**. Berlin: Springer, 2009, p.121-147.

COOPER, N.; MEGHANATHAN, N. Impact of Mobility Models on Multi-path Routing in Mobile Ad hoc Networks. **International Journal of Computer Networks and Communications**, [S.l.]: AIRCC, v.2, n.1, p.174-185, 2010.

CORMEN, T.H.; LEISERSON, C.E.; RIVEST R.L.; STEIN, C. **Introduction to Algorithms**, 2. ed. [S.l.]: MIT Press & McGraw-Hill, 2001.

CORTES, J.; MARTINEZ, S.; KARATAS T.; BULLO, F. Coverage control for mobile sensing networks. **IEEE Transactions on Robotics and Automation**. Washington, USA: IEEE Computer Society, v.20, i.2 p.243-255, 2004.

CURTIS, C.; LENZO, M.; MCCLURE M.; PREISS, B. The layered sensing operations center: a modeling and simulation approach to developing complex ISR networks. In

SPIE DEFENSE, SECURITY AND SENSING CONFERENCE, 2010, Orlando, USA. **Proceedings…**[S.l.:S.n.], v.7694, 2010, p.7694151-7694157.

DEMIRBAS, M.; SOYSAL O.; TOSUN, A.S. Data Salmon: A Greedy Mobile Basestation Protocol for Efficient Data Collection in Wireless Sensor Networks. In ASPNES, J.; SCHEIDELER, C.; ARORA, A.; MADDEN, S. (Eds.). IEEE INTERNATIONAL CONFERENCE ON DISTRIBUTED COMPUTING IN SENSOR SYSTEMS, 2007, Santa Fe, NM, USA. **Proceedings…** [S.l.]: Springer, 2007, p.267-280.

DIGI. XBee-PRO® 802.15.4 OEM RF Modules Product Manual. Product Specification. **DIGI**. 2007. Available in: <http://ftp1.digi.com/support/documentation/manual_xb_oem-rf-modules_802.15.4_v1.xAx.pdf> Accessed in: May. 2009.

DOHERTY, L.; GHAOUI L.E.; PISTER, K.S.J. Convex position estimation in wireless sensor networks. In IEEE INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS, 2001, Anchorage, AK, USA. **Proceedings…** Piscataway, USA: IEEE, 2001, p.1655–1663.

DORIGO, M.; DI, C.G. Ant Colony Optimization: A New Meta-Heuristic. In CONGRESS ON EVOLUTIONARY COMPUTATION, 1999, Washington DC, USA. **Proceedings…** Piscataway, USA: IEEE, 1999, p.1470-1477.

DRESSLER, F. **Self-Organization in Sensor and Actor Networks**. Erlangen: Wiley, 2007.

DRESSLER, F.; AKAN, O.B. Bio-inspired networking: from theory to practice. **IEEE Communications Magazine**. Washington, USA: IEEE Computer Society, v.48, i.11, p.176-183, 2010.

ERMAN, A.T.; HAVINGA, P. Data Dissemination of Emergency Messages in Mobile Multi-Sink Wireless Sensor Networks. In 9th IFIP ANNUAL MEDITERRANEAN AD HOC NETWORKING WORKSHOP, 2010, Juan Les Pins, France. **Proceedings…** [S.l.]: IEEE, 2010, p.1-8.

ERMAN, A.T.; HOESEL, L.; HAVINGA, P. AWARE: Platform for Autonomous Self-Deploying and Operation of Wireless Sensor-Actuator Networks Cooperating with AeRial Objects. In IEEE INTERNATIONAL WORKSHOP ON SAFETY, SECURITY AND RESCUE ROBOTICS, 2007, Rome, Italy. **Proceedings…** [S.l.]: IEEE Computer Society, 2007, p. 1-6.

ERMAN, A.T.; HOESEL, L.; HAVINGA, P. Enabling Mobility in Heterogeneous Wireless Sensor Networks Cooperating with UAVs for Mission-Critical Management. **IEEE Wireless Communications**. Washington, USA: IEEE Computer Society, v.15, i.6, p.38-46, 2008.

FIPA. Agent Communication Language Message Structure Specification. **Foundation for Intelligent Physical Agents**, Geneva, Switzerland, 2002. Available in: < http://www.fipa.org/specs/fipa00061/SC00061G.pdf > Accessed in: Mar. 2009.

FIPA. Agent Message Transport Service Specification. **Foundation for Intelligent Physical Agents**, Geneva, Switzerland, 2002. Available in: < http://www.fipa.org/specs/fipa00067/SC00067F.pdf > Accessed in: Mar. 2009.

FLAKE, G. W. **The computational beauty of nature: computer explorations of fractals, chaos, complex systems, and adaptation**. Cambridge, USA: MIT Press, 2000.

FOK, C.-L.; ROMAN, G.-C.; LU, C. Rapid Development and Flexible Deployment of Adaptive Wireless Sensor Network Applications. In 25th International Conference on Distributed Computing Systems, 2005, Columbus, USA. Washington, USA: IEEE Computer Society, 2005, p.653-662.

FOK, C.-L.; ROMAN, G.-C.; LU, C. Enhanced Coordination in Sensor Networks through Flexible Service Provisioning. In 11th INTERNATIONAL CONFERENCE ON COORDINATION MODELS AND LANGUAGES, 2009, Lisbon, Portugal. Lecture Notes in Computer Science 5521. **Proceedings…** [S.l.]: Springer-Verlag, pages 66-85, 2009.

FRANKLIN, S.; GRAESSER, A. Is it an agent, or just a program?: A taxonomy for autonomous agents. In Müller, J.; Wooldridge, M.; Jennings, N., (Eds.). **Intelligent Agents III Agent Theories, Architectures, and Languages**. 3rd International Workshop on Agent Theories, Architectures, and Languages, LNCS 1193. **Proceedings…** [S.l.]: Springer, 1997, p.21-35.

FREITAS, E. P.; ALLGAYER, R.S.; WEHRMEISTER, M. A.; PEREIRA, C. E.; LARSSON, T. Supporting Platform for Heterogeneous Sensor Network Operation based on Unmanned Vehicles Systems and Wireless Sensor Nodes. In IEEE INTELLIGENT VEHICLES SYMPOSIUM, 2009, Xi' an, China. **Proceedings…**Washington, USA: IEEE Intelligent Transport Systems Society, 2009a, p.786-791.

FREITAS, E. P.; FERREIRA, A. M.; PEREIRA C. E.; LARSSON, T. Middleware Support in Unmanned Aerial Vehicles and Wireless Sensor Networks for Surveillance Applications. In 3rd INTERNATIONAL SYMPOSIUM ON INTELLIGENT DISTRIBUTED COMPUTING, SCI 237, 2009, Ayia Napa, Cyprus. **Proceedings…**Berlin, Germany: Springer, 2009b, 289–296.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. An Agent Framework to Support Sensor Networks - Setup and

Adaptation. In IEEE INTERNATIONAL MULTICONFERENCE ON COMPUTER SCIENCE AND INFORMATION TECHNOLOGY, 2009, Mragowo, Poland. **Proceedings…**Washington, USA: IEEE Computer Society, 2009c, p. 533–540.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Evaluation of Coordination Strategies for Heterogeneous Sensor Networks Aiming at Surveillance Applications. In 8th ANNUAL IEEE CONFERENCE ON SENSORS, 2009, Christchurch, New Zealand. **Proceedings…**Washington, USA: IEEE Computer Society, 2009d, p.591–596.

FREITAS, E. P.; WEHRMEISTER, M. A.; PEREIRA, C. E.; FERREIRA, A. M.; LARSSON, T. Multi-Agents Supporting Reflection in a Middleware for Mission-Driven Heterogeneous Sensor Networks. In 3rd AGENT TECHNOLOGY FOR SENSOR NETWORKS (ATSN), IN CONJUNCTION WITH 8TH AUTONOMOUS AGENT AND MULTIAGENT SYSTEMS (AAMAS), 2009, Budapest, Hungary. **Proceedings…** [S.l.:s.n.], 2009e.

FREITAS, E. P.; HEIMFARTH, T.; ALLGAYER, R.S.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Coordinating Aerial Robots and Unattended Ground Sensors for Intelligent Surveillance Systems. **International Journal of Computers, Communications & Control**. [S.l.]:CCC Publications, v.5, n.1, p.52-70, 2010a.

FREITAS, E. P.; HEIMFARTH, T.; NETTO, I. F.; SÁ, A. G. C.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Enhanced Wireless Sensor Network Setup Strategy Supported by Intelligent Software Agents. In 9th ANNUAL IEEE CONFERENCE ON SENSORS, 2010, Waikoloa, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2010b, p.813-816.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Decentralized Task Distribution among Cooperative UAVs in Surveillance Systems Applications. In 7th IEEE/IFIP INTERNATIONAL CONFERENCE ON WIRELESS ON-DEMAND NETWORK SYSTEM AND SERVICES, 2010, Kranjska Gora, Slovenia. **Proceedings…** Piscataway, USA: IEEE Communication Society, 2010c, p.121-128.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Geo-Aware Handover of Mission Agents using Opportunistic Communication in VANET. In BALANDIN, S.; KOUCHERYAVY, Y.; HU, H. (Org.). Lecture Notes in Computer Science, V. 6294. 10th IEEE INTERNATIONAL CONFERENCE ON NEXT GENERATION WIRED/WIRELESS ADVANCED NETWORKING, 2010, St. Petersburg, Russia. **Proceedings…** 2010d, p.365-376.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Experimental Analysis of Coordination Strategies to Support Wireless Sensor Networks Composed by Static Ground Sensors and UAV-carried

Sensors. In IEEE INTERNATIONAL SYMPOSIUM ON PARALLEL AND DISTRIBUTED PROCESSING WITH APPLICATIONS, 2010, Taipei, Taiwan. **Proceedings…**Washington, USA: IEEE Computer Society, 2010e, p.152-161.

FREITAS, E. P.; HEIMFARTH, T.; COSTA, L. A. G.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Analyzing Different Levels of Geographic Context Awareness in Agent Ferrying over VANETs. In 26th ACM Symposium on Applied Computing (SAC'11), 2011, Taichung, Taiwan. **Proceedings…** New York, USA: ACM, 2011a, p.413-418.

FREITAS, E. P.; HEIMFARTH, T.; NETTO, I. F.; LINO, C. E.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Handling Failures of Static Sensor Nodes in Wireless Sensor Networks by Use of Mobile Sensors. In WORKSHOPS OF IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 2011, Biopolis, Singapore. **Proceedings…** [S.l.]: IEEE Computer Society, 2011b, p.127-134.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Multi-Agent Support in a Middleware for Mission-Driven Heterogeneous Sensor Networks. **The Computer Journal**. Oxford, UK: Oxford University Press, v. 54, i.3, p 406-420, Oxford Journals, 2011c.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Wireless Sensor Network Mission Setup Supported by Software Agents, Submitted to **ACM Transactions on Sensor Networks**. 2011d.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Bio-Inspired Networking to Support Cooperation among Wirelessly Connected Static and Mobile Sensor Nodes, Submitted to **IEEE/ACM Transactions on Networking**. 2011e.

FREW, E.W.; BROWN, T.X. Airborne Communication Networks for Small Unmanned Aircraft Systems. **Proceedings of the IEEE**. Washington, USA: IEEE Computer Society, v.96, i.12, p.2008-2027, 2008.

GABBAI, J.M.E.; YIN, H.; WRIGHT, W.A.; ALLINSON, N.M. Self-Organization, Emergence and Multi-Agent Systems. In INTERNATIONAL CONFERENCE ON NEURAL NETWORKS AND BRAIN, 2005, Beijing, China. **Proceedings…**[S.l.s.n.], 2005, p.1824-1863.

GALE, D. **The Theory of Linear Economic Models**. New York, USA: McGraw-Hill Book Company, Inc., 1960.

GAO, B.M.X. UAV Path Planning Based on Bidirectional Sparse A* Search Algorithm. In INTERNATIONAL CONFERENCE ON INTELLIGENT COMPUTATION

TECHNOLOGY AND AUTOMATION, 2010, Changsha, China, **Proceedings…** [S.l.:s.n.], 2010, p.1106-1109.

GAY, D.; LEVIS, P.; BEHREN, R.; WELSH, M.; BREWER, E.; CULLER, D. The nesC Language: A Holistic Approach to Networked Embedded Systems. In PROGRAMMING LANGUAGE DESIGN AND IMPLEMENTATION, 2003, San Diego, California, USA. **Proceedings…**[S.l.s.n.], 2003, p. 1-11.

GELERNTER, D. Generative Communication in Linda. **ACM Transactions on Programming Languages and Systems**. New York, USA: ACM, v.7, i.1, p.80–112, 1985.

GEORGAKARAKOU, C.E.; ECONOMIDES, A.A. Software Agent Technology: An Overview. In Tiako, P. F. (Ed.). **Software Applications: Concepts, Methodologies, Tools and Applications**. [S.l.]: IGI Global, 2009.

GERVASI, V.; PRENCIPE, G. Coordination without communication: the case of the flocking problem. **Discrete Applied Mathematics - Fun with Algorithms 2**. Amsterdam, The Netherlands: Elsevier Science Publishers, v. 144, i.3, p.324-344, 2004.

GIAMBERARDINO, P.; GABRIELE, S. Mobile sensors networks: a distributed solution to the area coverage problem. In 16th MEDITERRANEAN CONFERENCE ON CONTROL AND AUTOMATION, 2008, Ajaccio, France. **Proceedings…**[S.l.]: IEEE, 2008, p.1844-1849.

GIARRATANO, J.C.; RILEY, G. **Expert Systems: Principles and Programming**. Pacific Grove, CA, USA: Brooks/Cole Publishing Co., 1989.

GIORDANO, E.; FRANK, R.; GIOVANNI P.; GERLA, M. CORNER: a Realistic Urban Propagation Model for VANET. In 7th IEEE/IFIP INTERNATIONAL CONFERENCE ON WIRELESS ON-DEMAND NETWORK SYSTEM AND SERVICES, 2010, Kranjska Gora, Slovenia. **Proceedings…** Piscataway, USA: IEEE Communication Society, 2010, p.57-60.

GROCHOLSKY, B.; BAYRAKTAR, S.; KUMAR, V.; PAPPAS, G. UAV and UGV collaboration for active ground feature search and localization. In AIAA 3rd Unmanned Unlimited Technical Conference, 2004, Chicago, USA. **Proceedings…** [S.l.:s.n.], 2004.

GROSS, D.; HARRIS, C. M. **Fundamentals of Queueing Theory**. New York: Wiley, 1998.

GROSS, N. 21 ideas for the 21st century. **Business Week**. p.78–167, Aug. 30, 1999.

GURFIL, P.; KIVELEVITCH, E. Flock properties effect on task assignment and formation flying of cooperating unmanned aerial vehicles. **Proceedings of the**

Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering. [S.l.]: Sage Publications, v.221, i.3, p.410-416, 2007.

HANSEN, M. T.; JURDAK R.; KUSY, B. Unified Broadcast in Sensor Networks. In 10th ACM INTERNATIONAL CONFERENCE ON INFORMATION PROCESSING IN SENSOR NETWORKS, 2011, Chicago, IL, USA. **Proceedings…** [S.l.:s.n.], 2011, p.306-317.

HECTOR, A. A new classification scheme for software agents. In 3rd INTERNATIONAL CONFERENCE ON INFORMATION TECHNOLOGY AND APPLICATIONS, 2005, Sydney, Australia. **Proceedings…** Washington, USA: IEEE Computer Society, 2005, p.191-196.

HEIMFARTH T.; FREITAS, E.P. **GrubiX Simulator Statup Manual**. 2011. 37p. Technical Report, Federal University of Lavras, Lavras, Brazil, 2011.

HEIMFARTH, T.; FREITAS, E.P.; WAGNER, F.R.; LARSSON, T. Middleware Support for Wireless Sensor Networks: a Survey. In JIN H.; JIANG, W. (Eds.). **Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice**. [S.l.]:IGI Global, 2010a.

HEIMFARTH, T.; FREITAS, E. P.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Experimental Analysis of a Wireless Sensor Network Setup Strategy Provided by an Agent-oriented Middleware. In 24th IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 2010, Perth, Australia. **Proceedings…**Washington, USA: IEEE Computer Society, 2010b, p.820-826.

HEIMFARTH T.; JANACIK, P. Experiments with Biologically-Inspired Methods for Service Assignment in Wireless Sensor Networks. In Hinchey, M.; Pagnoni, A.; Rammig, F. J.; Schmeck; H. (Eds.). IFIP INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING - BIOLOGICALLY-INSPIRED COLLABORATIVE COMPUTING. **Proceedings…**Boston: Springer, 2008, p.71–84.

HEINZELMAN, W.B.; MURPHY, A.L.; CARVALHO, H.S.; PERILLO, M.A. Middleware to Support Sensor Network Applications. **IEEE Network**. New York, USA: IEEE Communication Society, v.18, i.1, p.6-14, 2004.

HENRICKSEN, K.; ROBINSON, R. A Survey of Middleware for Sensor Networks: State-of-the-Art and Future Directions. In ACM INTERNATIONAL WORKSHOP ON MIDDLEWARE FOR SENSOR NETWORKS, 2006, Melbourne, Australia. **Proceedings…** New York, USA: ACM, 2006, p.60-65.

HILL, J. **System Architecture for Wireless Sensor Networks**. 2003. 186 p. PhD thesis, Computer Science Division, University of California Berkeley, Berkeley, 2003.

HILL, J.; CULLER, D. **A wireless embedded sensor architecture for system-level optimization**. 2002. 12p. Technical Report, UC Berkeley, 2002.

HONG, J.; LU, S.; CHEN, D.; CAO, J. Towards Bio-Inspired Self-Organization in Sensor Networks: Applying the Ant Colony Algorithm. In IEEE 22nd INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 2008, Gino-wan City, Okinawa, Japan. **Proceedings…**Washington, DC, USA: IEEE Computer Society, 2008, p.1054-1061.

HWANG, K.; IN J.; EOM, D. S. Distributed Dynamic Shared Tree for Minimum Energy Data Aggregation of Multiple Mobile Sinks in Wireless Sensor Networks. In 3rd EUROPEAN WORKSHOP ON WIRELESS SENSOR NETWORKS, 2006, Zurich, Switzerland. **Proceedings…** [S.l.:s.n.], 2006, p.132-147.

IEEE. IEEE Standard 802.15.4. Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks. **Institute of Electrical and Electronics Engineers**. 2003. Available in: <http://standards.ieee.org/getieee802/help.html> Accessed in: Feb. 2009.

INTEL. Intel Mote 2 Engineering Platform Data Sheet. **Intel Coorporation**. 2006. Available in: <http://ubi.cs.washington.edu/files/imote2/docs/imote2-ds-rev2.0.pdf> Accessed in: Dec. 2011.

JAIN, S.; SHAH, R. C.; BRUNETTE, W.; BORRIELLO, G.; ROY, S. Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks. **Mobile Networks Applications**. Hingham, MA, USA: Kluwer Academic Publishers, v.11, i.3, p.327-339, 2006.

JIN, Y.; LIAO, Y.; MINAI A.A.; POLYCARPOU, M.M. Balancing Search and Target Response in Cooperative Unmanned Aerial Vehicle (UAV) Teams. **IEEE Transactions on System, Man, Cybernetics-Part B: Cybernetics**. [S.l.]: IEEE System, Man and Cybernetics Society, v.36, i.3, p.571-587, 2006.

JONG, K. A. D. **Evolutionary computation: a unified approach**. Cambridge, MA, USA: MIT Press, 2006.

JUANG, P.; OKI, H.; WANG, Y.; MARTONOSI, M.; PEH L.S.; RUBENSTEIN, D. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In 10th INTERNATIONAL CONFERENCE ON ARCHITECTURAL SUPPORT FOR PROGRAMMING LANGUAGES AND OPERATING SYSTEMS, 2002, San Jose, California, USA. **Proceedings…** New York, USA: ACM, 2002, p.96-107.

KAMINKA, G.A. Robots are Agents, Too! **AgentLink News**, p.16–17, 2004.

KARABOGA, D.; AKAY, B. A survey: algorithms simulating bee swarm intelligence. **Artificial Intelligence Review**. Norwell, MA, USA: Kluwer Academic Publishers, v.31, i.1-4, p.61–85, 2009.

KEWLANI, G.; IAGNEMMA, K. Mobility prediction for unmanned ground vehicles in uncertain environments. In SPIE Conference on Unmanned Systems, 2008, Orland, USA. **Proceedings…** Bellingham, USA: SPIE, 2008, p.69621G1–69621G12.

KHO, J.; ROGERS A.; JENNINGS, N. R. Decentralized control of adaptive sampling in wireless sensor networks. **ACM Transactions on Sensor Networks**. New York: ACM, v.5, i.3, p.1-35, 2009.

KIM, Y.; GU, D.; POSTLETHWAITE, I. Fault-Tolerant Cooperative Target Tracking In Distributed UAV Networks. In 17th WORLD CONGRESS IFAC CONFERENCE, 2008, Seoul, Korea. **Proceedings…** [S.l.]: Elsevier, 2008, p.8878-8883.

KLEINSCHMIDT, J. H. Genetic Algorithms for Wireless Sensor Networks. In DOPICO, J. R. R.; DORADO, J.; PAZOS, A., (Eds.). **Encyclopedia of Artificial Intelligence**. [S.l.]: IGI Global, p.755-758, 2009.

KOZIEROK, R.; MAES, P. A Learning Interface Agent for Scheduling Meetings. In of 1st INTERNATIONAL CONFERENCE ON INTELLIGENT USER INTERFACES, 1993, Orlando, Florida, USA. **Proceedings…**New York: ACM, 1993, p.81-88.

KOUSHANFAR, F.; POTKONJAK, M.; SANGIOVANNI-VINCENTELLI, A. Fault Tolerance in Wireless Ad-Hoc Sensor Networks. In IEEE SENSORS, 2002, Orlando, FL, USA. **Proceedings…** Washington, USA: IEEE Computer Society, v.2, 2002, p.1491-1496.

KUANG, L.; CAI, Z. Immune system based redeployment scheme for wireless sensor networks. In IET INTERNATIONAL CONFERENCE ON WIRELESS SENSOR NETWORK, 2010, Beijing, China. **Proceedings…**New York, USA: Curran Associates Inc, 2010, p.69-72.

KUMAR, R.; DAVE, M. Mobile Agent as an Approach to Improve QoS in Vehicular Ad Hoc Network. **International Journal of Computer Applications, Special Issue on "Mobile Ad-hoc Networks" MANETs**. [S.l.:s.n.]v.2, p.67-72, 2010.

KUMAR, P.; GUNES, M.; MUSHTAQ Q.; SCHILLER, J. Optimizing Duty-Cycle for Delay and Energy Bound WSN Applications. In 24th IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS WORKSHOPS, 2010, Perth, Australia. **Proceedings…** Washington, USA: IEEE Computer Society, 2010, p.692-697.

KUNERT, K. **Architectures and Protocols for Performance Improvements of Real-Time Networks**. 2010. 300 p. Ph.D. Thesis - Chalmers University of Technology, Göteborg, Sweden, 2010.

KUORILEHTO, M.; HÄNNIKÄINEN, M.; HÄMÄLÄINEN, T.D. A survey of application distribution in wireless sensor networks. **EURASIP Journal on Wireless Communications and Networking**. New York, USA: Hindawi Publishing Corp., v.2005, i.5, p.774-788, 2005.

LABELLA, T.H.; DRESSLER, F. A Bio-Inspired Achitecture for Division of Labour in SANETs. In 1st IEEE/ACM INTERNATIONAL CONFERENCE ON BIO-INSPIRED MODELS OF NETWORK, INFORMATION AND COMPUTING SYSTEMS, 2006, Cavalese, Italy. **Proceedings…** New York: ACM, 2006, 8p.

LABONTE, G. Canadian artic Sovereignty: Local intervention by flocking UAVs. In 2nd IEEE INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE FOR SECURITY AND DEFENSE APPLICATIONS, 2009, Ottawa, Canada. **Proceedings…** Washington, USA: IEEE Computer Society, 2009, p.110-117.

LACEY, N.; HEXMOOR, H. Norm Adaptation and Revision in a Multi-Agent System. In 16th INTERNATIONAL FLORIDA ARTIFICIAL INTELLIGENCE RESEARCH SOCIETY CONFERENCE, 2003, St. Augustine, Florida, USA. **Proceedings…** [S.l.,s.n.], 2003, p.27-31.

LANGE, D. B.; OSHIMA, M. Seven Good Reasons for Mobile Agents. **Communications of the ACM**. New York: ACM, v.42, i.3, p.88-89, 1999.

LANGENDOEN, K.; BAGGIO, A.; VISSER, O. Murphy loves potatoes: experiences from a pilot sensor network deployment in precision agriculture. In 20th INTERNATIONAL PARALLEL AND DISTRIBUTED PROCESSING SYMPOSIUM, 2006, Rhodes Island, Greece. **Proceedings…** Washington, USA: IEEE Computer Society, 2006, p.174-181.

LEE, S.H.; LEE, S.; SONG, H.; LEE, H.S. Wireless sensor network design for tactical military applications: remote large-scale environments. In 28th IEEE CONFERENCE ON MILITARY COMMUNICATIONS, 2009, Boston, USA. **Proceedings…**Piscataway, NJ, USA: IEEE Press, 2009a, p.911-917.

LEE, U.; MAGISTRETTI, E.; GERLA, M.; BELLAVISTA P.; CORRADI, A. Dissemination and Harvesting of Urban Data Using Vehicular Sensing Platforms. IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY. New York: IEEE, v.58, i.2, p.882-901, 2009b.

LESSMANN, J.; JANACIK, P.; LACHEV, L.; ORFANUS, D. Comparative Study of Wireless Network Simulators. In 7th INTERNATIONAL CONFERENCE ON

NETWORKING, 2008, Cancun, Mexico. **Proceedings…** Washington, USA: IEEE Computer Society, 2008, p.517-523.

LESSMANN, J.; HEIMFARTH T.; JANACIK, P. ShoX: An Easy to Use Simulation Platform for Wireless Networks. In 10th INTERNATIONAL CONFERENCE ON COMPUTER MODELING AND SIMULATION, 2008, Cambridge, England. **Proceedings…** Washington, USA: IEEE Computer Society, 2008, p.410-415.

LEVIS, P.; CULLER, D. E. Maté: a Tiny Virtual Machine For Sensor Networks. In 10th INTERNATIONAL CONFERENCE ON ARCHITECTURAL SUPPORT FOR PROGRAMMING LANGUAGES AND OPERATING SYSTEMS, 2002, San Jose Fairmont, CA, USA. **Proceedings…** New York: ACM, 2002, p.85-95.

LEVIS, P.; MADDEN, S.; POLASTRE, J.; SZEWCZYK, R.; WHITEHOUSE, K.; WOO, A.; GAY, D.; HILL, J.; WELSH, M.; BREWER, E.; CULLER, D. TinyOS: An operating system for wireless sensor networks. **Ambient Intelligence**. Berlin: Springer-Verlag, p.115-148, 2004.

LI, C.; ZHAI, L.; SUN, L. WMOS: A Wireless Message-Oriented System for Wireless Sensor Networks. In WASE INTERNATIONAL CONFERENCE ON INFORMATION ENGINEERING, 2010, Beidaihe, China. **Proceedings…**Washington, USA: IEEE Computer Society, 2010, p.300-303.

LIDSTRÖM, K.; LARSSON, T. A Spatial QoS Requirements Specification for V2V Applications. In 2010 IEEE INTELLIGENT VEHICLES SYMPOSIUM, 2010, San Diego, CA, USA. **Proceedings…**Piscataway, NJ, USA: IEEE, 2010, p.548-553.

LIN, C.; HE, Y-X.; XIONG, N. An Energy-Efficient Dynamic Power Management in Wireless Sensor Networks. In 5th INTERNATIONAL SYMPOSIUM ON PARALLEL AND DISTRIBUTED COMPUTING, 2006, Timisoara, Romania. **Proceedings…** Washington, USA: IEEE Computer Society, 2006, p.148-154.

LIU, T.; MARTONOSI, M. Impala: A Middleware System for Managing Autonomic, Parallel Sensor Systems. In 9th ACM SIGPLAN SYMPOSIUM ON PRINCIPLES AND PRACTICE OF PARALLEL PROGRAMMING, 2003, San Diego, California, USA. **Proceedings…** New York: ACM, 2003, p.107-118.

LIU, Y.; YU, F. Immunity-based intrusion detection for wireless sensor networks. In IEEE INTERNATIONAL JOINT CONFERENCE ON NEURAL NETWORKS, IEEE WORLD CONGRESS ON COMPUTATIONAL INTELLIGENCE, 2008, Hong Kong, China. **Proceedings…** [S.l.,s.n.], 2008, p.439-444.

MACHADO, M.V.; GOUSSEVSKAIA, O.; MINI, R.A.F.; REZENDE, C.G.; LOUREIRO, A.A.F.; MATEUS, G.R.; NOGUEIRA, J.M.S. Data Dissemination in

Autonomic Wireless Sensor Networks. **IEEE Journal on Selected Areas in Communications**. [S.l.]: IEEE Communication Society, v.23, i.12, p.2305-2319, 2005.

MADDEN, S.R.; FRANKLIN, M.J.; HELLERSTEIN, J.M.; HONG, W. TinyDB: an acquisitional query processing system for sensor networks. **ACM Transaction on Database Systems- Special Issue: SIGMOD/PODS**. New York: ACM, v.30, i.1, p.122-173, 2005.

MALONE, T.W.; CROWSTON, K. What is Coordination Theory and How Can It Help Design Cooperative Work Systems? In CONFERENCE ON COMPUTER SUPPORTED COOPERATIVE WORK, 1990, Los Angeles, California, USA. **Proceedings…** New York: ACM, 1990, p.357-370.

MALONE, T.W.; CROWSTON, K. The interdisciplinary study of coordination. **ACM Computing Surveys**. New York: ACM, v26, i.1, p.87-119, 1994.

MCQUIDDY, J.H. A roadmap for future UGS. In SPIE Defense, Security and Sensing Conference, 2010, Orlando, USA. **Proceedings…** Bellingham, USA: SPIE, 2010, p.76940N1-76940N12.

MEYER, H.; HUMMEL, K.A. A geo-location based opportunistic data dissemination approach for MANETs. In 4th ACM Workshop on Challenged Networks, 2009, Beijing, China. **Proceedings…** New York: ACM, 2009, p.1-8.

MIKROKOPTER. MikroKopter: modular multicopter systems. 2011. **MikroKopter** Available in: <http:// http://www.mikrokopter.de> Accessed in Aug. 2011.

MINI, R. A. F.; LOUREIRO, A. A. F. Energy in Wireless Sensor Networks. In Garbinato, B.; Miranda, H.; Rodrigues, L. (Eds.). **Middleware for Network Eccentric and Mobile Applications**. Berlin, Germany: Springer-Verlag, 2009, p.3-24.

MINI, R. A. F.; MACHADO, M. V.; LOUREIRO, A. A. F.; NATH, B. Prediction-based energy map for wireless sensor networks. **Ad Hoc Network Journal**. Amsterdam, The Netherlands: Elsevier, v.3, i.2, p.235-253, 2005.

MITTAL, N.; PANGHAL, A.; CHAUHAN, R. S.; ARYA, S. K. Ant Colony Based Algorithm for Routing in MANETs. **Journal of Communication and Computer**. Libertyville, USA: David Publishing, v.7 i.12, p.44-47, 2010.

MODI, P.; SHEN, W.; TAMBE, M.; YOKOO, M. Adopt: Asynchronous distributed constraint optimization with quality guarantees. **Artificial Intelligence**. Essex, UK: Elsevier Science Publishers, v.161, p.149–180, 2005.

MORRIS, A.; GIORGINI P.; ABDEL-NABY, S. Simulating BDI-based Wireless Sensor Networks. In 2009 IEEE/WIC/ACM INTERNATIONAL JOINT CONFERENCE ON WEB INTELLIGENCE AND INTELLIGENT AGENT

TECHNOLOGY, 2009, Milano, Italy. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p.78-81.

MSB. Submeter-scale aircraft. 2011. **MSB Co**. Available in <http://spyplanes.com> Accessed in Fev. 2010.

MULDOON, C. **An Agent Framework for Ubiquitous Services**. 2008. 349 p. Ph.D. Thesis - School of Computer Science and Informatics, University College Dublin, Ireland, 2008.

MUNIR, S. A.; REN, B.; JIAO, W.; WANG, B.; XIE, D.; MA, J. Mobile Wireless Sensor Network: Architecture and Enabling Technologies for Ubiquitous Computing. In 21st International Conference on Advanced Information Networking and Applications Workshops, 2007, Niagara Falls, Ontario, Canada. **Proceedings…**Washington, USA: IEEE Computer Society, 2007, p.113-120.

MUTAMBARA, A.G.O. **Decentralized Estimation and Control for Multisensor Systems**. Boca Raton, USA: CRC Press LLC, 1998.

NAKAMURA, E.F.; LOUREIRO, A. A. F.; FRERY, A. C. Information fusion for wireless sensor networks: Methods, models, and classifications. **ACM Computing Surveys**. New York: ACM, v.39, i.3, p.9-es, 2007.

NICULESCU, D.; NATH, B. Localized positioning in ad hoc networks. **Ad Hoc Networks**. [S.l.]: Elsevier, v.1, i.2-3, p.247–259, 2003.

NICULESCU, D.; NATH, B. Trajectory-based forwarding and its applications. In 9th ANNUAL INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING, 2003, San Diego, CA, USA. **Proceedings…** New York: ACM, 2003, p.260–272.

NORTH, M. J.; MACAL, C. M. Foundations of and Research Advances in Artificial Life Modeling with Repast 3 and Repast Simphony. In KOMOSINSKI, M.; ADAMATZKY, A. (Eds.). **Artificial Life Models in Software**. Berlin, Germany: Springer, 2009, p.37-60.

OBJECT TECHNOLOGY INTERNATIONAL. Eclipse Platform: Technical Overview. Version 2.1. **Object Technology International, Inc**. 2003. Available in: <http://www.eclipse.org/whitepapers/eclipse-overview.pdf> Accessed in: Jan. 2009.

OCHIAI, H.; ISHIZUKA, H.; KAWAKAMI Y.; ESAKI, H. A Field Experience on DTN-Based Sensor Data Gathering in Agricultural Scenarios. In IEEE SENSORS 2010, 2010, Waikoloa, HI, USA. **Proceedings…** Washington, USA: IEEE Computer Society, 2010, p.955-958.

O'HARE, G.M.P. Agent Factory: An Environment for the Fabrication of Multi-Agent Systems. In O'Hare, G.M.P.; Jennings, N. (Eds.). **Foundations of Distributed Artificial Intelligence**. New York, USA: John Wiley and Sons Inc, p. 449-484, 1996.

OLFATI-SABER, R. Flocking for multi-agent dynamic systems: algorithms and theory. **IEEE Transactions on Automatic Control**. Piscataway, NJ, USA: Control Systems Society of IEEE, v.51, i.3, p.401-420, 2006.

OLIVEIRA, H.A.B.F.; BARRETO, R.S.; FONTAO, A.L.; LOUREIRO, A.A.F.; NAKAMURA, E.F. A Novel Greedy Forward Algorithm for Routing Data Toward a High Speed Sink in Wireless Sensor Networks. In 19th INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS, 2010, Zurich, Switzerland. **Proceedings…** New York: IEEE Computer Society, 2010, p.1-7.

ORDOWER, R.; DIXON, L.; LYNCH, N. Collaborative air/ground command and control for responsive persistent ISR operations using unmanned systems. In SPIE DEFENSE, SECURITY AND SENSING CONFERENCE, 2010, Orlando, USA. **Proceedings…** Bellingham, USA: SPIE, 2010, p.76940S1- 76940S11.

ORHAN, D.; ILKER, K.; FATIH T.; KAYHAN, E. A Survey of Agent Technologies for Wireless Sensor Networks. **IETE Technical Review**. [S.l.]: IETE, v.28, i.2, p.168-184, 2011.

PARKINSON B.; SPILKER, J. **Global Positioning System: Theory and Application**. Washington, USA: American Institute of Astronautics and Aeronautics, 1996.

PARPINELLI, R.S.; LOPES, H.S. New inspirations in swarm intelligence: a survey. **International Journal of Bio-Inspired Computing**. Genève, Switzerlan: Inderscience Publishers, v.3, i.1, p.1-16, 2011.

PARTAP, U. The Pollination Role of Honeybees. In Hepburn H.R.; Radloff, S.E., (Eds.). **Honeybees of Asia**. Berlin, Germany: Springer-Verlag, p.227-255, 2011.

POLASTRE, J.; HILL J.; CULLER, D. Versatile Low Power Media Access for Wireless Sensor Networks. In ACM SenSys'04, 2004, Baltimore, USA. **Proceedings…** New York, USA: ACM, 2004, p.95-107.

POPA, D.O.; MYSOREWALA, M.F.; LEWIS, F.L. Deployment Algorithms and In-Door Experimental Vehicles for Studying Mobile Wireless Sensor Networks. **ACIS International Journal of Sensor Networks**. Genève, Switzerlan: Inderscience Publishers, v.6, i.1, p.28-43, 2009.

PRUSINKIEWICZ, P.; HANAN, J. **Lindenmayer Systems, Fractals and Plants**. Secaucus, NJ, USA: Springer-Verlag New Your Inc, 1989.

QUARITSCH, M.; KRUGGL, K.; WISCHOUNIG-STRUCL, D.; BHATTACHARYA, S.; SHAH, M.; RINNER, B. Networked UAVs as aerial sensor network for disaster management applications. **E & I Elektrotechnik und Informationstechnik**. Berlin, Germany: Springer, v.127, n.3, p.56-63, 2010.

RAJAGOPALAN, R.; VARSHNEY, P.K. Data aggregation techniques in sensor networks: A survey. **IEEE Communications Surveys & Tutorials**. Washington, USA: IEEE Computer Society, v.8, i.4, p.48-63, 2006.

RAO, A. S. AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In 7th EUROPEAN WORKSHOP ON MODELLING AUTONOMOUS AGENTS IN A MULTI-AGENT WORLD, 1996, Eindhoven, The Netherlands. **Proceedings…** Secaucus, NJ, USA: Springer-Verlag New York Inc, 1996, p.42-55.

RAS, E.; BECKER M.; KOCH, J. Engineering Tele-Health Solutions in the Ambient Assisted Living Lab. In 21st INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS WORKSHOPS (AINAW'07), 2007, Niagara Falls, Ontario, Canada. **Proceedings…** Washington, USA: IEEE Computer Society, 2007, p.804-809.

RAO, A. S.; GEORGEFF, M. P. BDI agents: from theory to practice. In 1st INTERNATIONAL CONFERENCE ON MULTIAGENT SYSTEMS, 1995, San Francisco, USA. **Proceedings…**Menlo Park, CA, USA: AAAI Press, 1995, p.312-319.

RAUF, A.; ZAFAR, M.A.; ASHRAF, Z.; AKHTAR, H. Aerodynamic modeling and state-space model extraction of a UAV using DATCOM and Simulink. In 3rd INTERNATIONAL CONFERENCE ON COMPUTER RESEARCH AND DEVELOPMENT, 2011, Shanghai, China. **Proceedings…**Washington, USA: IEEE Computer Society, 2011, p.88-92.

REN, B.; MA, J.; CHEN, C. The hybrid mobile wireless sensor networks for data gathering. In 2006 INTERNATIONAL CONFERENCE ON WIRELESS COMMUNICATIONS AND MOBILE COMPUTING, 2006, Vancouver, British Columbia, Canada. **Proceedings…** New York, USA: ACM, 2006, p.1085-1090.

ROSS, S.M. **Introduction to Probability Models**. 10. ed. San Diego, CA, USA: Academic Press, 2007.

RUSSELL, S.J.; NORVIG, P. **Artificial Intelligence: A Modern Approach**. 2. ed. Upper Saddle River, New Jersey: Prentice Hall, 2003.

SAKARKAR, G.; SHELKE, N.M. A new classification scheme for autonomous software agent. In International Conference on Intelligent Agent & Multi-Agent Systems, 2009, Chennai, India. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p.1-2.

SCERRI, P.; OWENS, S.; SYCARA K.; LEWIS, M. User evaluation of a GUI for controlling an autonomous persistent surveillance team. In SPIE DEFENSE, SECURITY AND SENSING CONFERENCE, 2010, Orlando, USA. **Proceedings…** Bellingham, USA: SPIE, 2010, p.76940E1-76940E12.

SCHURR, N.; MARECKI, J.; LEWIS, J.P.; TAMBE, M.; SCERRI, P. The Defacto System: Coordinating Human-Agent Teams for the Future of Disaster Response. **MULTI-AGENT PROGRAMMING Multiagent Systems, Artificial Societies, and Simulated Organizations**. New York, USA: Springer Science+Business Media Inc, v.15, i.3, p.197-215, 2005.

SHEHORY, O.; SYCARA, K.; CHALASANI, P.; JHA, S. Agent Cloning: An Approach to Agent Mobility and Resource Allocation. **IEEE Communications Magazine**. New York, USA: IEEE Communication Society, v.36, i.7, p.58–67, 1998.

SHEN, C.; SRISATHAPORNPHAT, C.; JAIKAEO, C. Sensor Information Networking Architecture and Applications. **IEEE Personal Communications**. New York, USA: IEEE Communication Society, v.8, i.4, p.52-59, 2001.

SINGH, C.P.; VYAS, O.P.; TIWARI, M.K. A Survey of Simulation in Sensor Networks. In INTERNATIONAL CONFERENCE ON COMPUTATIONAL INTELLIGENCE FOR MODELLING CONTROL & AUTOMATION, 2008, Vienna, Austria. **Proceedings…**Washington, USA: IEEE Computer Society, 2008, p.867-872.

SKYDRONES. Skydrones Tecnologia Aviônica Ltda. 2001. **Skydrones**. Available in: <http://www.skydrones.com.br> Accessed in: August 2011.

SOLTANI, S.; MISRA, K.; RADHA, H. On link-layer reliability and stability for wireless communication. In 14th ACM INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING, 2008, San Francisco, USA. **Proceedings…** New York, USA: ACM, 2008, p.327-338.

SOUTO, E. GUIMARAES; G. VASCONCELOS; G. VIEIRA; M. ROSA, N.; FERRAZ, C. A message-oriented middleware for sensor networks. In 2nd WORKSHOP ON MIDDLEWARE FOR PERVASIVE AND AD-HOC COMPUTING, 2004, Toronto, Canada. **Proceedings…** New York, USA: ACM, v.77, 2004, p.127-134.

SOUZA, L. M. S.; VOGT, H.; BEIGL, M. **A Survey on Fault Tolerance in Wireless Sensor Networks**. 2007. 11 p. Interner Bericht. Fakultät für Informatik, Universität Karlsruhe, Karlsruhe, Germany, 2007.

SPYROPOULOS, T.; PSOUNIS, K.; RAGHAVENDRA, C. S. Performance Analysis of Mobility-assisted Routing. In 7th ACM INTERNATIONAL SYMPOSIUM ON MOBILE AD HOC NETWORKING AND COMPUTING, 2006, Florence, Italy. **Proceedings…** New York, USA: ACM, 2006, p.49-60.

STOJMENOVIC, I. Position-Based Routing in Ad Hoc Networks. **IEEE Communications Magazine**. New York, USA: IEEE Communication Society, v.40, i.7, p.128-134, 2002.

STRANDERS, R.; ROGERS, A.; JENNINGS, N. A Decentralized, On-line Coordination Mechanism for Monitoring Spatial Phenomena with Mobile Sensors. In 2nd INTERNATIONAL WORKSHOP ON AGENT TECHNOLOGY FOR SENSOR NETWORKS, 2008, Estoril, Portugal. **Proceedings…** [S.l.,s.n.], 2008, 9-15.

SUN. Sun™ SPOT Programmer's Manual. Release v6.0 (Yellow). **Sun Microsystems Laboratories / Oracle**, 2010. Available in: <http://sunspotworld.com/docs/Yellow/SunSPOT-Programmers-Manual.pdf>
Accessed in: Jun 2011.

STANSBURY, R. S.; VYAS, M. A.; WILSON, T. A. A Survey of UAS Technologies for Command, Control, and Communication (C3). **Journal of Intelligent & Robotic Systems**. Hingham, MA, USA: Kluwer Academic Publishers, v.54, i.1-3, p.61-78, 2009.

TAHERKORDI, A.; TALEGHAN, M. A.; SHARIFI, M. Dependability Considerations in Wireless Sensor Networks Applications. **Journal of Networks**. Oulu, Finland: Academy Publisher, v.1, i.6, p.28-35, 2006.

TANENBAUM, A. S. **Computer Networks**. 4. ed. Upper Saddle River, NJ, USA: Prentice Hall, 2003.

TANENBAUM, A. S.; STEEN, M. **Distributed systems: principles and paradigms**. Upper Saddle River, NJ, USA: Prentice Hall, 2007.

TAN, S.K.; MUNRO, A. Adaptive Probabilistic Epidemic Protocol for Wireless Sensor Networks in an Urban Environment. In 16th INTERNATIONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS, 2007, Honolulu, HI, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2007, p.1105-1110.

TEI, K.; YOSHIOKA, N.; FUKAZAWA, Y.; HONIDEN, S. Geographically Bound Mobile Agent in MANET. In 2nd ANNUAL INTERNATIONAL CONFERENCE ON MOBILE AND UBIQUITOUS SYSTEMS: NETWORKING AND SERVICES, 2005, San Diego, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2005, p.516-518.

TOLLE, G.; POLASTRE, J.; SZEWCZYK, R.; CULLER, D.; TURNER, N.; TU, K.; BURGESS, S.; DAWSON, T.; BUONADONNA, P.; GAY D.; HONG, W. A Macroscope in the Redwoods. In 3rd INTERNATIONAL CONFERENCE ON

EMBEDDED NETWORKED SENSOR SYSTEMS, 2005, San Diego, USA. **Proceedings…** New York, USA: ACM, 2005, p.51–63.

TYNAN, R.; O'HARE, G.M.P.; RUZZELLI, A.G. Multi-agent system methodology for wireless sensor networks. **Multiagent Grid Systems**. Amsterdam, The Netherlands: IOS Press, v.2, p.491–503, 2006.

UVS. **2009/2010 UAS-Yearbook UAS: The Global Perspective**. 7. ed. UVS International. Paris, France: Blyenburgh & Co, 2009.

UYSAL-BIYIKOGLU, E.; PRABHAKAR, B.; GAMAL, A. Energy-efficient packet transmission over a wireless link. **IEEE/ACM Transactions on Networking**. Piscataway, NJ, USA: IEEE Press, v.10, i.4, p.487-499, 2002.

VINYALS, M.; RODRIGUEZ-AGUILAR, J. A.; CERQUIDES, J. A Survey on Sensor Networks from a Multiagent Perspective. **The Computer Journal**. Oxford, UK: Oxford University Press, v.54, i.3, p.455-470, Oxford Journals, 2011.

VLASSIS, N. **A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence**. [S.l.]: Morgan & Claypol Publishers, 2007.

WALD, L. Definitions and terms of reference in data fusion. INTERNATIONAL ARCHIVES OF PHOTOGRAMMETRY AND REMOTE SENSING, 1999, Valladolid, Spain. **Proceedings…** [S.l.]: ISPRS Society, v.32, 1999, p.7-4-3 W6.

WANG, Y.; ATTEBURY, G.; RAMAMURTHY, B. A Survey of Security Issues In Wireless Sensor Networks**. IEEE Communications Surveys & Tutorials**. New York, USA: IEEE Communication Society, v.8, i.2, p.2-23, 2006.

WANG, Z.; TIANFIELD, H.; JIANG, P. A Framework for Coordination in Multi-Robot Systems. In IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL INFORMATICS, 2003, Banff, Alberta, Canada. **Proceedings…**Washington, USA: IEEE Computer Society, 2003, p.483-489.

WHITMAN, E. C. SOSUS: The "Secret Weapon" of Undersea Surveillance. **Undersea Warfare**. [S.l.:s.n.]v.7, n.2, 2005.

WOOLDRIDGE, M. **An Introduction to MultiAgent Systems**. Chichester, UK: John Wiley & Sons Ltd, 2002.

WOOLDRIDGE, M. Agent-Based Software Engineering. **IEE Proceedings Software Engineering**. [S.l.:s.n.], v.144, i.1, p.26-37, 1997.

WOOLDRIDGE M.; JENNINGS, N. R. Agent Theories, Architectures, and Languages: a Survey. In WOOLDRIDGE M.; JENNINGS, N. R. (Eds.). **Intelligent Agents**. Berlin: Springer-Verlag, p.1-22, 1995.

WU, H.; FUJIMOTO, R.M.; RILEY, G.F.; HUNTER, M. Spatial Propagation of Information in Vehicular Networks. **IEEE Transactions on Vehicular Technology**. Piscataway, NJ, USA: Vehicular Technology Society, v.58, i.1, p.420-411, 2009.

WU, H.; PALEKAR, M.; FUJIMOTO, R.M.; LEE, J.; KO, J.; GUENSLER, R.; HUNTER, M. Vehicular networks in urban transportation systems. In 2005 NATIONAL CONFERENCE ON DIGITAL GOVERNMENT RESEARCH, 2005, Atlanta, Georgia, USA. **Proceedings…** [S.l.]: Digital Government Society of North America, 2005, p.9-10.

XIAO, Y.; ZHANG, Y. Surveillance and Tracking System with Collaboration of Robots, Sensor Nodes, and RFID tags. In 18th INTERNATONAL CONFERENCE ON COMPUTER COMMUNICATIONS AND NETWORKS, 2009, San Francisco, CA, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p.1-5.

XU, N. A survey of sensor network applications. 2003. 9 p. Technical report - Computer Science Department, University of Southern California, 2003.

XUAN, P.; LESSER, V.; ZILBERSTEIN, S. Communication Decisions in Multi-agent Cooperation: Model and Experiments. In 5th INTERNATIONAL CONFERENCE ON AUTONOMOUS AGENTS, 2001, Montreal, Quebec, Canada. **Proceedings…** New York, USA: ACM, 2001, p.616-623.

YARVIS, M.; KUSHAFNAGAR, N.; SINGH, H.; RANGARAJAN, A.; LIN, Y.; SINGH, S. Exploiting heterogeneity in sensor networks. In 24th Annual Joint Conference of the IEEE Computer and Communications Societies, 2005, Miami, FL, USA. **Proceedings…**Piscataway, NJ, USA: IEEE, v. 2, 2005, p.878–890.

YICK, J.; MUKHERJEE, B.; GHOSAL, D. Wireless sensor network survey. **Computer Networks**. New York, NY, USA: Elsevier, v.52, i.12, p.2292-2330, 2008.

YU, Y.; KRISHNAMACHARI, B.; PRASANNA, V. K. Issues in Designing Middleware for Wireless Sensor Networks. **IEEE Network**. New York, USA: IEEE Communication Society, v.18, i.1, p.15- 21, 2005.

YUE, W.; SUN, Q.; JIN, S. Performance Analysis of Sensor Nodes in a WSN with Sleep/Wakeup Protocol. In 9th SYMPOSIUM ON OPERATIONS RESEARCH AND ITS APPLICATIONS, 2010, Chengdu-Jiuzhaigou, China. **Proceedings…** [S.l.]: ORSC & APORC, 2010, p.370–377.

ZHANG, L. DUAN, L. QIAN Z.; HUANG, G. WSN Node Localization Technology Based on Genetic Algorithm. **Computer Engineering.** [S.l.]: Jisuanji Gongcheng, v.36, i.10, p.85-87, 2010.

ZHANG, Y.; XIAO, Y. Primate-inspired Scent Marking for Mobile and Static Sensors and RFID Tags. In 18th Internatonal Conference on Computer Communications and Networks, 2009, San Francisco, CA, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p.1-5.

ZHAO, W.; AMMAR M.; ZEGURA, E. A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks. In 5th ACM INTERNATIONAL SYMPOSIUM ON MOBILE AD HOC NETWORKING AND COMPUTING, 2004, Tokyo, Japan. **Proceedings…** New York, USA: ACM, 2004, p.187-198.

ZHAO, F.; GUIBAS, L. **Wireless Sensor Networks - An Information Processing Approach**. S. Francisco, USA: Morgan Kaufman Publisher, 2004.

ZHOU, Y.; FANG, Y.; ZHANG, Y. Securing Wireless Sensor Networks: A Survey. **IEEE Communications & Surveys**. New York, USA: IEEE Communication Society, v.10, i.3, p.6-28, 2008.

ZHOU, B.; XU, K.; GERLA, M. Group and swarm mobility models for ad hoc network scenarios using virtual tracks. In IEEE MILCOM, 2004, Monterey, CA, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2004, p.289–294.

APPENDIX A LIST OF PUBLICATIONS AND AWARDS

A.1 Award

Santander Universities Award - category Science and Innovation: Information Technology and Communication, for the SUAVIC (Sensors and Unmanned Aerial Vehicles Integration and Coordination) project. The award was offered by the Santander Bank. November 2010.

A.2 Main Publications in Scope of the Thesis

FREITAS, E. P.; HEIMFARTH, T.; COSTA, L. A. G.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Analyzing Different Levels of Geographic Context Awareness in Agent Ferrying over VANETs. In 26th ACM Symposium on Applied Computing (SAC'11), 2011, Taichung, Taiwan. **Proceedings…** New York, USA: ACM, 2011, p.413-418.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Multi-Agent Support in a Middleware for Mission-Driven Heterogeneous Sensor Networks. **The Computer Journal**. Oxford, UK: Oxford University Press, v. 54, i.3, p 406-420, Oxford Journals, 2011.

FREITAS, E. P.; HEIMFARTH, T.; NETTO, I. F.; SÁ, A. G. C.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Enhanced Wireless Sensor Network Setup Strategy Supported by Intelligent Software Agents. In 9th ANNUAL IEEE CONFERENCE ON SENSORS, 2010, Waikoloa, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2010, p.813-816.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Geo-Aware Handover of Mission Agents using Opportunistic Communication in VANET. In BALANDIN, S.; KOUCHERYAVY, Y.; HU, H. (Org.). Lecture Notes in Computer Science, V. 6294. 10th IEEE INTERNATIONAL CONFERENCE ON NEXT GENERATION WIRED/WIRELESS ADVANCED NETWORKING, 2010, St. Petersburg, Russia. **Proceedings…** 2010, p.365-376.

HEIMFARTH, T.; FREITAS, E. P.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Experimental Analysis of a Wireless Sensor Network Setup Strategy Provided by an Agent-oriented Middleware. In 24th IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 2010, Perth, Australia. **Proceedings…**Washington, USA: IEEE Computer Society, 2010, p.820-826.

HEIMFARTH, T.; FREITAS, E. P.; WAGNER, F. R.; LARSSON, T. Middleware Support for Wireless Sensor Networks: a Survey. In: Jin, H.; Jiang, W. (Orgs). **Handbook of Research on Developments and Trends in Wireless Sensor Networks: From Principle to Practice**. Hershey, Information Science Reference (IGI Global), March 2010.

FREITAS, E. P.; HEIMFARTH, T.; ALLGAYER, R.S.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Coordinating Aerial Robots and Unattended Ground Sensors for Intelligent Surveillance Systems. **International Journal of Computers, Communications & Control**. [S.l.]:CCC Publications, v.5, n.1, p.52-70, 2010.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Decentralized Task Distribution among Cooperative UAVs in Surveillance Systems Applications. In 7th IEEE/IFIP INTERNATIONAL CONFERENCE ON WIRELESS ON-DEMAND NETWORK SYSTEM AND SERVICES, 2010, Kranjska Gora, Slovenia. **Proceedings…** Piscataway, USA: IEEE Communication Society, 2010, p.121-128.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Evaluation of Coordination Strategies for Heterogeneous Sensor Networks Aiming at Surveillance Applications. In 8th ANNUAL IEEE CONFERENCE ON SENSORS, 2009, Christchurch, New Zealand. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p.591–596.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. An Agent Framework to Support Sensor Networks - Setup and Adaptation. In IEEE INTERNATIONAL MULTICONFERENCE ON COMPUTER SCIENCE AND INFORMATION TECHNOLOGY, 2009, Mragowo, Poland. **Proceedings…**Washington, USA: IEEE Computer Society, 2009, p. 533–540.

FREITAS, E. P.; FERREIRA, A. M.; PEREIRA C. E.; LARSSON, T. Middleware Support in Unmanned Aerial Vehicles and Wireless Sensor Networks for Surveillance Applications. In 3rd INTERNATIONAL SYMPOSIUM ON INTELLIGENT DISTRIBUTED COMPUTING, SCI 237, 2009, Ayia Napa, Cyprus. **Proceedings…**Berlin, Germany: Springer, 2009, 289–296.

FREITAS, E. P.; ALLGAYER, R.S.; WEHRMEISTER, M. A.; PEREIRA, C. E.; LARSSON, T. Supporting Platform for Heterogeneous Sensor Network Operation

based on Unmanned Vehicles Systems and Wireless Sensor Nodes. In IEEE INTELLIGENT VEHICLES SYMPOSIUM, 2009, Xi' an, China. **Proceedings…**Washington, USA: IEEE Intelligent Transport Systems Society, 2009, p.786-791.

FREITAS, E. P.; WEHRMEISTER, M. A.; PEREIRA, C. E.; FERREIRA, A. M.; LARSSON, T. Multi-Agents Supporting Reflection in a Middleware for Mission-Driven Heterogeneous Sensor Networks. In 3rd AGENT TECHNOLOGY FOR SENSOR NETWORKS (ATSN), IN CONJUNCTION WITH 8TH AUTONOMOUS AGENT AND MULTIAGENT SYSTEMS (AAMAS), 2009, Budapest, Hungary. **Proceedings…** [S.l.:s.n.], 2009.

A.3 Publications Representing Secondary Contributions

FREITAS, E. P.; BÖSCH, B.; ALLGAYER, R. S.; STEINFELD, L.; WAGNER, F. R.; CARRO, L.; PEREIRA, C. E.; LARSSON, T. Mobile Agents Model and Performance Analysis of a Wireless Sensor Network Target Tracking Application. In: S. BALANDIN, Y. KOUCHERYAVY, AND H. HU. (Org.). 10th IEEE INTERNATIONAL CONFERENCE ON NEXT GENERATION WIRED/WIRELESS ADVANCED NETWORKING, 2011, St. Petersburg, Russia. **Proceedings…** Berlin, Germany: Springer, 2011, p.274-286.

FREITAS, E. P.; HEIMFARTH, T.; NETTO, I. F.; LINO, C. E.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Handling Failures of Static Sensor Nodes in Wireless Sensor Networks by Use of Mobile Sensors. In WORKSHOPS OF IEEE INTERNATIONAL CONFERENCE ON ADVANCED INFORMATION NETWORKING AND APPLICATIONS, 2011, Singapore. **Proceedings…**Washington, USA: IEEE Computer Society, 2011, p.127 – 134.

MOTTER, P.; ALLGAYER, R. S.; MÜLLER, I.; PEREIRA, C. E.; FREITAS, E. P. Practical issues in Wireless Sensor Network localization systems using received signal strength indication. In IEEE SENSORS APPLICATIONS SYMPOSIUM, 2011, San Antonio, TX, USA. **Proceedings…**Washington, USA: IEEE Computer Society, 2011, p.227-232.

FREITAS, E. P.; HEIMFARTH, T.; NETTO, I. F.; LINO, C. E.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. UAV Relay Network to Support WSN Connectivity. In IEEE INTERNATIONAL CONGRESS ON ULTRAMODERN TELECOMMUNICATIONS AND CONTROL SYSTEMS, 2010, Moscow, Russia. **Proceedings…**Washington, USA: IEEE Computer Society, 2010, p.309-314.

FREITAS, E. P.; HEIMFARTH, T.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Experimental Analysis of Coordination Strategies to Support Wireless Sensor Networks Composed by Static Ground Sensors and UAV-carried

Sensors. In IEEE INTERNATIONAL SYMPOSIUM ON PARALLEL AND DISTRIBUTED PROCESSING WITH APPLICATIONS, 2010, Taipei, Taiwan. **Proceedings…**Washington, USA: IEEE Computer Society, 2010, p.152-161.

FREITAS, E. P.; HEIMFARTH, T.; WEHRMEISTER, M. A.; WAGNER, F. R.; FERREIRA, A. M.; PEREIRA, C. E.; LARSSON, T. Using Link Metric to Improve Communication Mechanisms and Real-time Properties in an Adaptive Middleware for Heterogeneous Sensor Networks. In ISA CONFERENCE - FIRST INTERNATIONAL WORKSHOP ON MOBILE & WIRELESS NETWORKS, LNCS 5576, 2009, Seoul, Korea. **Proceedings…**Berlin, Germany: Springer, 2009. p. 422-431.

FREITAS, E. P.; BINOTTO, A. P. D.; PEREIRA, C. E.; STORK, A.; LARSSON, T. Dynamic reconfiguration of tasks applied to an UAV system using aspect orientation. EEE INTERNATIONAL SYMPOSIUM ON PARALLEL AND DISTRIBUTED PROCESSING WITH APPLICATIONS, 2008, Sydney, Australia. **Proceedings…**Washington, USA: IEEE Computer Society, 2008, p.292-300.

APPENDIX B ACADEMIC ACTIVITIES

B.1 Supervision

Bachelor Thesis

Yin Jin. Digital Jukebox. 2011. (Bachelor in Datateknik) - Halmstad University. Role: Main Supervisor.

Denny Antonio Toazza and Tae Hyun Kim. Navigation Control of an Unmanned Aerial Vehicle (UAV). 2010. (Bachelor in Elektroteknik) - Halmstad University. Role: Main Supervisor.

Danqing Ni and Lin Ge. A MOD Player for GBA. 2010. (Bachelor in Datorsystemteknik) - Halmstad University. Role: Main Supervisor.

Chaoyou Dai; Yifei Li and Weiming Zhai. Communication among UAVs. 2010. (Bachelor in Datateknik) - Halmstad University. Role: Main Supervisor.

Ugur Bozkurt. Assembly of a UAV: hardware design of a UAV. 2009. (Bachelor in Elektroteknik) - Halmstad University. Role: Main Supervisor.

Master Thesis

Daniel Kifetew Shenkutie and Prashanth Kumar Patil Shinde. Residual Energy Monitoring in Wireless Sensor Networks. 2011. (Master in Embedded and Intelligent Systems) - Halmstad University. Role: Main Supervisor.

Thomas Josef Lampoltshammer; Stefan Plank and Thomas Nowotny. ICT System Design & Implementation Using Wireless Sensors to Support Elderly In-home Assistance. 2011. (Master in Nätverksteknik) - Halmstad University. Role: Main Supervisor.

Samar Sajadian and Alia Ibrahim. Wireless Sensor Network Group Connectivity. 2010. (Master in Nätverksteknik) - Halmstad University. Role: Co-Supervisor.

Javed Iqbal and Farhan Moughal. Wireless Sensor Network Setup: Wireless sensor motes embedded programming. 2010. (Master in Nätverksteknik) - Halmstad University. Role: Co-Supervisor.

B.2 Teaching

Course: Distributed Real-time Systems Laboratory, Halmstad University, 2009, 2010 and 2011.

B.3 Service to the Academic Community

Program Committee Member

2009 IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA), 2009.

Baltic Congress on Future Internet and Communications (BCFIC '11). 2011.

2nd International Workshop on Recent Advances in Broadband Access Networks (RABAN'11). 2011.

Organizer Committee Member

Halmstad – Örebro – Skövde PhD Students Colloquium - IT Workshop. 2010.

APPENDIX C TRANSLATION OF THE FIRST CHAPTER

1 Introdução

1.1 Considerações Preliminares

Redes de sensores sem fio (RSSFs) vêm ganhando importância nos últimos anos devido ao número de aplicações nas quais tal tecnologia pode ser usada. Esta tecnologia foi apontada pela revista Business Week (GROSS, 1999) como uma das 21 mais importantes tecnologias para o século 21. Os avanços e a miniaturização de computação, equipamentos sensores e de comunicação, fornecem meios para o desenvolvimento de nós sensores baratos, porém inteligentes, que podem ser usados em diferentes configurações de rede. Essas redes são capazes de prover uma variedade de dados, num maior ou menor grau de pre-processamento, a sistemas de informação centralizados com diferentes aplicativos (AKYILDIZ et al., 2002).

As primeiras aplicações de redes de sensores apareceram na área de sistemas militares, como o SOSUS (Sound Surveillance System), que era um arranjo de sensores acústicos instalados pela marinha norte americana no fundo do oceano para detectar submarinos soviéticos durante a guerra fria (WHITMAN, 2005). Estudos realizados pela DARPA (Defence Advanced Research Projects Agency) patrocinaram um grande número de projetos militares que envolviam tecnologia de RSSF (CHONG; KUMAR, 2003). Naquele tempo, a maioria das redes de sensores eram conjuntos de sensores conectados por fios. Com os avanços nas radio comunicações, nós sensores foram equipados com radio transceptores, e o uso de redes de sensores conectados sem fio criou a possibilidade de se usar tal tecnologia em diversas novas aplicações (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). Essas novas possibilidades chamaram a atenção de diversas comunidades de pesquisa, tanto no meio acadêmico quanto na indústria, que têm aplicado e experimentado redes de sensores sem fio em uma grande variedade de aplicações como monitoramento de vida selvagem (LIU; MARTONOSI, 2003), sistemas de apoio em habitações inteligentes (RAS; BECKER; KOCH, 2007), assistência em missões de resgate em casos de desastres (ERMAN; HOESEL; HAVINGA, 2008), prevenção e controle de fogo (FOK; ROMAN; LU, 2005), dentre outras.

Uma classe de sistemas baseados em RSSFs que tem aplicação tanto no meio militar quanto civil é o monitoramento de área. Sistemas de monitoramento podem ser usados em regiões de fronteira, ou ao longo de perímetros, ou para cobrir áreas. Com este uso eles podem ser aplicados, por exemplo, no controle de território, monitoramento de tráfego em estradas, prevenção e monitoramento de fogo em matas e

regiões remotas, monitoramento de linhas de transmissão de energia elétrica e segurança, como em sistemas de monitoramento de mercadorias em regiões portuárias (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005) (XU, 2003). Uma característica importante que este tipo de sistema deve ser capaz de prover é a flexibilidade de realizar diferentes tipos de missão de sensoriamento de acordo com necessidades específicas de seus usuários. Isto é importante devido a diferentes requisitos de aplicação, bem como a natureza dinâmica do ambiente no qual tais sistemas são empregados, nos quais mudanças podem ocorrer a qualquer momento. Tais mudanças não devem interferir no desempenho do sistema, que deve continuar a executar suas missões. Também é importante reparar que tais sistemas podem ser empregados na execução de mais de várias missões simultaneamente na área na qual são implantados. Isto requer flexibilidade na configuração dos nós sensores de acordo com as informações de interesses de seus usuários. Ao mesmo tempo, os nós sensores devem ser capazes de se auto-configurarem e formarem uma rede de modo que consigam cumprir as diferentes missões que são requisitadas.

Dependendo da aplicação e das necessidades específicas das missões, sistemas de monitoramento precisam de diferentes tipos de nós sensores, os quais realizam aquisição de uma grande variedade de dados brutos que são então tratados e refinados. O resultado deste processo é representado por informação de mais alto nível sobre o fenômeno observado o qual é finalmente entregue aos usuários finais (BARDELABEN, 2003). Porém, a capacidade de uma rede de sensores realizar este trabalho com sucesso depende da cooperação entre os diferentes sensores disponíveis na rede, de tal forma que eles possam contribuir no alcance do objetivo comum da rede. Esta cooperação é particularmente desafiadora quando além da diferença entre os tipos de dados que os sensores são capazes de prover, eles também diferem em relação a outras capacidades, tais como mobilidade (AKYILDIZ; KASIMOGLU, 2004).

Mobilidade é uma capacidade importante que permite o reposicionamento de nós sensores (GIAMBERARDINO; GABRIELE, 2008), que pode ser provida através da montagem dos sensores em veículos autônomos como veículos aéreos ou terrestres não-tripulados (VANTs ou VTNTs) (KIM; GU; POSTLETHWAITE, 2008) (POPA; MYSOREWALA; LEWIS, 2009). Outras alternativas são redes de sensores móveis compostas por nós que cooperam de maneira mais oportunista, como as compostas por dispositivos portáteis como celulares ou PDAs (TEI et al., 2005), ou mesmo veículos convencionais circulando em áreas urbanas (LEE et al., 2009b).

Uma abordagem interessante é o uso cooperativo de ambos sensores estáticos e móveis (AKYILDIZ; KASIMOGLU, 2004), o qual é uma nova tendência de particular interesse para sistemas de monitoramento (ERMAN; HOESEL; HAVINGA, 2008). Esta combinação promissora permite a implementação de sistemas de monitoramento compostos de sensores estáticos simples e baratos, os quais podem ser empregados em grande número de unidades, cooperando com outros poucos móveis, que são mais sofisticados, porém mais caros.

Além da preocupação com o comportamento funcional que deve ser endereçado pela cooperação acima mencionada, sendo composta por sensores estáticos, móveis ou ambos tipos de sensores, RSSFs tem diversos outros requisitos que devem ser levados em consideração.

Sensores estáticos são geralmente limitados em recursos de processamento, memória e energia. Sendo assim, mecanismos de cooperação em RSSFs devem ser simples, implementados por algoritmos de baixa complexidade e que utilizem pouco espaço para armazenar dados. Além disto, considerando-se toda interação entre sensores se dá via comunicação sem fio e observando que este tipo de comunicação é a tarefa

mais cara em termos de consumo de energia (MINI; LOUREIRO, 2009), a cooperação só pode ser considerada eficiente se utiliza de racional a comunicação entre os nós sensores, portanto economizando recursos de energia.

Para os nós móveis, geralmente não se observa as mesmas limitações em termos de poder de processamento, memória disponível e recursos energéticos, considerando que eles são embarcados em veículos suficientemente grandes. Porém, devido à dinamicidade da topologia da rede causada pela movimentação dos nós, a comunicação com outros nós da rede não é confiável, o que é uma preocupação que deve ser considerada (LEE et al., 2009b). Ademais, dependendo da aplicação, sigilo é uma grande preocupação e o uso desnecessário de comunicação sem fio pode expor os sistema a entidades hostis, o que é o caso de aplicações militares (LEE et al., 2009a).

Observando estes requisitos, este trabalho objetiva endereçar o problema de cooperação entre sensores estáticos e móveis conectados por rede sem fio, diminuindo a necessidade de comunicação entre os nós. As estratégias propostas para lidar com o problema são baseadas em mecanismos com inspiração biológica e agentes de software móveis. A proposta considera sistemas de monitoramento como o principal cenário de motivação, o qual é usado para suportar a formulação do problema. Porém, é importante ressaltar que as técnicas desenvolvidas são genéricas o suficiente para serem aplicadas em outros cenários de aplicação.

Depois da apresentação dos cenários de motivação, a formulação dos problemas específicos tratados neste trabalho é apresentada, seguida da descrição dos objetivos, delimitação do escopo, e contribuições alcançadas. Finalmente, apresenta-se a metodologia do trabalho, e conteúdo do restante da tese.

1.2 Motivação

A demanda por novos produtos por um lado e avanços tecnológicos por outro lado suportam um ao outro num cadeia fechada na qual as demandas vindas do mercado estimulam o avanço tecnológico, enquanto novas tecnologias criam possibilidades para a criação de novos produtos (ADNER; LEVINTHAL, 2001). Este laço de iteração “sem fim” para inovação não é novo (ADNER; LEVINTHAL, 2001); ele tem funcionado por vários anos em diferentes setores onde tecnologias são desenvolvidas e empregadas, e o mesmo pode ser observado no desenvolvimento de tecnologias de RSSF e suas aplicações (GROSS, 1999). Este mecanismo de inovação na área de RSSFs trouxe recentemente para a realidade diversas aplicações, e muitas são esperadas para o futuro próximo (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005).

Em diferentes domínios, desde cuidados médicos (ALEMDAR; ERSOY, 2010) até sistemas militares (LEE et al., 2009a), a variedade de aplicações de RSSFs é muito grande. Neste trabalho o foco se concentra em sistemas de monitoramento, que tem utilidade tanto em aplicações militares quanto civis. Controle de fronteiras, monitoramento de vida selvagem, operações de resgate após desastres, monitoramento de tráfico em rodovias, operações de garantia da lei e da ordem e segurança são algumas das possíveis aplicações, para se mencionar poucas.

Sistemas de monitoramento têm por objetivo detectar eventos de interesse pré-definidos, que podem representar perigo ou ameaça, como a presença de veículos ou pessoas não autorizadas numa determinada área, ou ocorrência de eventos catastróficos,

como fogo ou inundação, dependendo da aplicação (ORDOWER; DIXON; LYNCH, 2010) (XU, 2003).

Um sistema de monitoramento de propósito geral deve apresentar necessidades específicas que motivam o uso de sensores estáticos, móveis e ambos os tipos. A escolha pelo tipo de sensor para um determinado propósito tem diferentes razões, como o sigilo do sistema, ou a inadequação de um dos tipos devido a limitações decorrentes do ambiente operacional. Logo, é muito provável que apenas sensores estáticos, apenas móveis, ou a combinação de ambos seja desejável em vários sistemas de monitoramento, dependendo das circunstâncias nas quais estes sistemas são usados (CURTIS et al., 2010). Logo, eles devem oferecer estas diferentes possibilidades aos seus usuários.

Sensores estáticos terrestres (MCQUIDDY, 2010), variam de simples e baratas plataformas como sensores piezoelétricos, comumente usados em RSSFs (AKYILDIZ et al., 2002), até sensores mais sofisticados e mais caros como sensores de imagem, como câmeras de infra-vermelho (MCQUIDDY, 2010). A despeito de sua sofisticação, em geral esses sensores tem limitações quanto ao recurso de energia, uma vez que são alimentados por baterias que devem durar o máximo de tempo possível, devido a questões práticas de reposição ou recarga, como o fato de serem empregados em ambientes hostis ou perigosos. O potencial do uso de tais sensores é aumentado quando eles são conectados em rede, o que possibilita a extração de informações semanticamente mais ricas, através do uso de técnicas de fusão de dados (NAKAMURA; LOUREIRO; FRERY, 2007).

A mobilidade dos nós sensores permite uma utilização avançada destes sensores, uma vez que eles são capazes de se reposicionar de acordo com as necessidades das missões de monitoramento. Isto permite que sensores de imagem cubram vastas áreas e com isto aumentando a faixa de atuação destes sensores se comparados à situação na qual eles são estáticos (GIAMBERARDINO; GABRIELE, 2008). Sendo móveis no solo ou no ar, esta capacidade de trocar a posição do sensor prove um aprimoramento importante no desempenho de missões de monitoramento, e o uso de vários sensores móveis interligados em rede é uma extensão natural desta abordagem (GROCHOLSKY et al., 2004), que tem sido massivamente explorada em sistemas de monitoramento militar (BARDELABEN, 2003).

A combinação de sensores estáticos e móveis em uma rede integrada é uma abordagem promissora que permite o desenvolvimento de avançados sistemas de monitoramento (ERMAN; HOESEL; HAVINGA, 2008) (XIAO; ZHANG, 2009). Esta combinação de sensores tem sua motivação baseada na complementaridade de características que eles provêm. Por um lado, sensores estáticos terrestres simples usualmente empregados em RSSFs são geralmente baratos, o que permite seu uso em grande número. No entanto, eles são apenas capazes de prover dados simples, básicos, como detecção de presença ou de movimento (AKYILDIZ et al., 2002). Por outro lado, típicos sensores móveis são mais caros e sofisticados, como radares, câmeras de luz visível ou de infravermelho, montados em plataformas móveis como carros (LEE et al., 2009b), robôs ou aeronaves que podem se mover em duas (POPA; MYSOREWALA; LEWIS, 2009) ou três (KIM; GU; POSTLETHWAITE, 2008) dimensões. Além disto, a mobilidade prove a característica singular mencionada acima de tornar possível a cobertura de vastas áreas geográficas (GIAMBERARDINO; GABRIELE, 2008).

Além dos benefícios específicos de cada tipo de sensor, estáticos e móveis, eles também apresentam desvantagens específicas (YICK; MUKHERJEE; GHOSAL, 2008). Uma RSSF composta apenas de nós estáticos simples não é capaz de prover informações tão ricas quanto às fornecidas por sensores mais sofisticados. No entanto, uma RSSF composta apenas por sensores sofisticados, sejam eles estáticos ou móveis, talvez não seja possível de ser instalada onde é necessária e talvez apresente custos proibitivos, dependendo do quão sofisticados tais sensores sejam, e o quão caras sejam as plataformas móveis utilizadas. A combinação de sensores estáticos e móveis pode superar a limitação dos dados providos pelos sensores estáticos simples, bem como os altos custos associados aos sofisticados sensores móveis [YZ09]. Isto é possível através do uso dos sensores estáticos para disparar o deslocamento ou reposicionamento dos móveis para áreas onde eles são necessários, o que pode ser feito através da criação de envio de alarmes com requisições dos sensores móveis para as áreas indicadas nos alarmes. Com este tipo de abordagem, um número reduzido de sensores móveis pode ser empregado, o que diminui o custo total do sistema, enquanto mantem sua capacidade de adquirir dados semanticamente mais ricos.

Usando apenas sensores estáticos, apenas móveis, ou combinações de ambos os tipos, o cenário de monitoramento pode ser descrito como uma vasta área na qual regiões menores representam áreas de interesse para diferentes usuários do sistema. Esses usuários podem submeter diferentes missões de sensoriamento que devem ser executadas separadamente nas diferentes áreas de interesse para adquirir dados sobre determinado fenômeno, e, portanto tais áreas são definidas como áreas de missão (MA). A Figura 1.1 apresenta uma visão geral deste cenário, no qual três Mas são definidas.

Como pode-se observar na Figura 1.1, as MAs podem ser de diferentes formatos e dimensões. Na figura, três exemplos de formatos para MAs são apresentados, um circular, um retangular, e outro quadrado. Além disto, essas MAs podem ter sobreposições entre si. Nesses três exemplos ilustrados na figura, as MAs tem diferentes tipos de sensores executando monitoramento em seus limites. MA-1 tem uma RSSF composta apenas de nós estáticos, enquanto MA-2 tem apenas nós móveis, e MA-3 tem os dois tipos de sensores. Por exemplo, considerando-se sensores sendo carregados por VANTs como sensores móveis, MA-1 pode ser uma zona proibida para voo, logo, apenas sensores estáticos podem ser usados para o monitoramento. Outra possibilidade é o caso no qual carros são usados como sensores móveis e MA-1 é uma região montanhosa sem infraestrutura de estradas. MA-2 pode ser uma região na qual o uso de sensores estáticos é evitado devido restrições de sigilo, por exemplo. Este é o caso no qual sensor móveis camuflados no meio de outros veículos dos quais são indistinguíveis podem ser usados, como apresentado em Mobieyes (LEE et al., 2009b), no qual uma rede de sensores veiculares (VSN) é usada para prover monitoramento urbano. Na MA-3 ambos os tipos, sensores móveis e estáticos são combinados para executar missões de monitoramento. Cada uma das MAs apresentadas na Figura 1.1 representa um sub-cenário de monitoramento com suas próprias restrições e peculiaridades que permitem ou requerem a utilização de diferentes tipos de sensores, ou a sua combinação.

Os usuários de tais sistemas de monitoramento especificam missões de sensoriamento através da especificação do tipo de informação na qual estão interessados. Geralmente linguagens de alto nível são desejáveis para a especificação de

missões, como linguagens de consulta (MADDEN et al., 2005), possivelmente incluindo interfaces gráficas de interface com o usuário (GUI) para facilitar a utilização do sistema. Realmente a usabilidade de sistemas de monitoramento é um grande problema, uma vez que quantidades enormes de dados são geradas por eles e requerem processamento e apresentação apropriados de forma que possam ser realmente úteis (SCERRI et al., 2010). A especificação de uma missão deve conter no mínimo parâmetros tais como a localização de onde ela deve ser executada, i.e. a MA, os limites temporais para sua execução e o tipo e quantidade de dados que devem ser colhidos.

Parâmetros adicionais podem ser especificados, dependendo das possibilidades de configuração oferecidas pelo sistema. De acordo com tais possibilidades e do tipo de sensores que compõe o sistema, usuários podem especificar, por exemplo: formações a serem seguidas pelos sensores móveis (BEARD et al., 2006), padrões de movimento ou caminhos a serem seguidos (GAO 2010), fusão (NAKAMURA; LOUREIRO; FRERY, 2007) ou agregação (RAJAGOPALAN; VARSHNEY, 2006) de dados, limitações temporais (BEARD et al., 2006), entre outros.

1.2.1 Cooperação entre Nós Sensores Estáticos

O primeiro cenário se refere à MA-1 apresentada na Figura 1.1 e prove um exemplo de um caso ordinário de RSSF estática usada para monitoramento de área (KUORILEHTO; HÄNNIKÄINEN; HÄMÄLÄINEN, 2005). Um grande número de sensores estáticos são espalhados na MA, de acordo com um determinado padrão, que pode ser aleatório ou uniforme seguindo um determinado padrão. Os sensores se comunicam entre si através de enlaces sem fio, dentro de um raio configurável, porém limitada, de comunicação. Devido à característica de difusão na comunicação sem fio, todos os nós dentro do raio de transmissão de um nó transmissor recebem as mensagens enviadas.

Tais sensores podem realizar medições simples, como diferença no campo magnético e concentração no nível de CO_2, dentre outras, que podem indicar a ocorrência de eventos de interesse, como a presença de veículos, pessoas ou fogo. Várias técnicas de fusão e agregação de dados podem ser usadas para extrair informações dos dados brutos coletados por esses sensores (NAKAMURA; LOUREIRO; FRERY, 2007). Como resultado, informações de alto nível podem ser entregues aos usuários finais, representando o resultado das missões por eles submetidas.

Os usuários especificam missões que são enviadas para a RSSF. Uma vez que essas missões chegam a RSSF através de um ponte de acesso, elas devem ser disseminadas através da rede e alocadas aos sensores de modo a serem executadas. Dependendo da localização do ponto de acesso em relação a MA, as missões têm que ser retransmitidas por outros sensores até chegar a suas respectivas MA. As missões devem ser informadas a um número suficiente de sensores dentro de suas respectivas MA, de modo que o número requerido de sensores necessários para a sua execução sejam informados sobre elas, e possam ser alocados na sua execução de acordo com os requisitos da missão.

Este processo, desde a especificação da missão até o engajamento dos sensores na sua execução, pode ser chamado setup da missão o qual pode ilustrado na sequência apresentada na Figura 1.2.

A Figura 1.2a ilustra um usuário especificando uma missão de sensoriamento, a qual é processada e enviada para o ponto de acesso a rede. Figura 1.2b apresenta a disseminação da missão; desde o ponto de acesso até atingir os nós na MA, informando-os sobre a missão. Finalmente, na Figura 1.2c destaca-se os nós sensores alocados para executar a missão na MA.

1.2.2 Cooperação entre Nós Sensores Móveis

O segundo cenário está relacionado a MA-2, que apresenta exclusivamente sensores móveis, constituindo uma RSSF móvel. Neste cenário os sensores se movem na área bem como para fora dela. Enquanto se movem dentro da MA-2, eles executam a respectiva missão. Quando eles deixam a área, eles tentam transferir a missão para algum outro sensor que esteja se movendo para dentro da área.

Considerando um cenário simplificado no qual dois sensores tem o objetivo principal de cobrir uma área, se movendo de acordo com padrões de movimento pré-definidos, eles podem executar concorrentemente uma missão de sensoriamento específica numa determinada MA. Como seu objetivo principal não é esta missão, eles podem transferi-la entre si de acordo com seu movimento, de tal forma que a missão seja assumida pelo sensor que esteja se movendo para dentro da MA. A Figura 1.3 apresenta a situação na qual dois sensores patrulham uma área na qual uma área menor (MA) é definida e tem uma missão para ser executada dentro de seus limites. O nó sensor S-1 patrulha seguindo um movimento de oeste para leste, enquanto o sensor S-2 executa um movimento de norte para sul. A Figura 1.3a mostra a situação na qual o sensor S-1está carregando e executando a missão dentro de MA, o que é graficamente representado por sua cor cinza. Na Figura 1.3b, S-1 está se movendo para fora de MA e se comunica com o nó S-2, executando a transferência da missão para este nó. A Figura 1.3c apresenta a situação em que S-2 assume a missão, denotado pela cor cinza de seu símbolo, e está começando a executá-la ao entrar na MA.

Como os sensores não tem, em princípio, nenhum padrão de movimento pré-estabelecido que pudesse facilitar a transferência de missões, a rede formada por eles para executar monitoramento de área pode ser considerada uma rede oportunísticas (CONTI et al., 2009). Neste tipo de rede, desconexões e reconexões são frequentes, e os nós oportunisticamente conseguem entregar suas mensagens quando se encontram. Neste cenário, um nó sensor carregando a missão que deixa a MA pode não encontrar nenhum nó para o qual possa transferir a missão. Logo, este nó deve executar esta transferência mais tarde quando encontrar outro nó fora da MA. Este nó por sua vez pode ele mesmo executar a missão, se ele entrar na MA, ou transferir para outro nó no caso de trocar de direção para outra localidade fora da MA. Este sequencia de transferências de missão entre pares de nós podem ser vistas como a migração da missão de um local para outro no cenário como um todo, no qual o nó que hospeda a missão está localizado para outro que esteja dentro da MA.

A densidade de nós na MA e nas suas redondezas é de grande importância na maneira como as interações entre os nós se dará, bem como quão frequentes elas ocorrerão. Importante observar que o exemplo apresentado na Figura 1.3 é apenas uma ilustração de como uma transferência de missão se dá. De fato, a utilidade desta

cooperação entre nós móveis é mais facilmente percebida em redes maiores com maior número de nós, nas quais a migração de missões através de sequencias de transferências delas entre vários nós pode ser observada.

Com respeito à densidade de nós, este cenário tem um grande potencial para aplicações de monitoramento urbano. Neste tipo de ambiente, uma rede de sensores veiculares (VSN) pode ser usada para monitorar pontos de obstrução de tráfego, níveis de poluição ou para coletar dados para prevenir ataques terroristas (LEE et al., 2009b). Outra possibilidade é a formação de redes com nós heterogêneos nas quais veículos e telefones celulares podem ser usados para monitorar níveis de poluição acústica dentro de determinada MA. Em cenários de resgate após desastres, estas redes heterogêneas podem ser usadas para realizar missões de sensoriamento que permitam a aquisição de dados que podem ser usados por equipes de resgate que estejam respondendo situações de emergência (TEI et al., 2005). Devido a este grande número de possíveis aplicações que se enquadram neste segundo cenário, o monitoramento urbano realizado por sensores embarcados em veículos é o cenário selecionado para ser explorado como parte desta tese.

1.2.3 Cooperação entre Nós Sensores Estáticos e Móveis

O terceiro cenário, relacionado a MA-3, combina a utilização de ambos os tipos de sensores, estáticos e móveis. Considerando o monitoramento de vastas áreas usando RSSFs compostas de sensores móveis e estáticos, a escolha do tipo de sensor móvel é tema importante [YZ09].

Plataformas móveis terrestres, como VTNTs, têm a habilidade de mover os sensores para locais onde eles são necessários, mas tem-se uma reduzida área de operação devido tanto a particularidades do relevo do terreno, quanto a obstáculos, mesmo existindo abordagens que endereçam tais problemas (KEWLANI; IAGNEMMA, 2008). Plataformas móveis no ar se movem em três dimensões (ou apenas em duas a uma determinada altura), como VANTs (QUARITSCH et al, 2010), podem oferecer melhores resultados. Estas plataformas tem a capacidade de mover os sensores para o local desejado pelo alto, evitando assim obstáculos que podem ser encontrados no solo. Mas mesmo no caso em que VANTs são usados, a escolha deve ser feita entre diferentes plataformas VANT. Plataformas VANT de pequeno porte, como as produzidas pela MLB (MSB, 2011), são muito mais baratas que plataformas VANT de grande porte, como a Predator e o Globalhawk (STANSBURY; VYAS; WILSON, 2009). Plataformas VANT de pequeno porte possibilitam o uso não apenas de alguns, mas de várias unidades no sistema, aumentando a sua capacidade e melhorando sua robusticidade através de redundância. Ademais, plataformas VANT de pequeno porte possibilitam o sistema realizar missões em regiões sensíveis, nas quais plataformas VANT de grande porte não tem acesso, como ambientes urbanos (FREW; BROWN, 2008). Baseado nesses argumentos, o interesse por plataformas VANT de pequeno porte se justifica, e por isto são consideradas no estudo deste cenário específico.

Neste cenário, considera-se a situação na qual alguns VANTs sobrevoam uma MA, seguindo um caminho aleatório ou um padrão de movimento determinado. Eles são equipados com sensores sofisticados, como câmeras de luz visível, infravermelho, e radares SAR/ISAR, enquanto nós terrestres estáticos equipados com sensores mais

simples são espalhados pela MA. Os sensores dos VANTs são capazes de prover informações mais detalhadas se comparados aos sensores estáticos terrestres. O número de VANTs é, no entanto, muito menor que o número de sensores terrestres estáticos, e a ideia é fazer com que eles trabalhem de maneira cooperativa, de tal forma q complementem uns aos outros, como já discutido. Assume-se que a distribuição dos nós sensores estáticos terrestres garante a cobertura de toda a área, e que quando observam e identificam um evento de possível interesse, eles emitem alarmes que são enviados aos VANTs, e.g. pedindo que estes executem observações mais detalhadas com seus sensores mais sofisticados.

Similarmente ao primeiro cenário, os sensores se comunicam via enlaces sem fio dentro de um raio, o qual permite todos os nós dentro do raio de comunicação de um nó transmissor receber a mensagem enviada.

O comportamento do sistema é definido da seguinte forma. Os sensores estáticos terrestres são configurados para detectar eventos de interesse, que possam indicar algum tipo de ameaça ou perigo, o que é definido por níveis limites relacionado aos valores medidos. Quando as medições atingem tais níveis, o critério de detecção é atingido. Na ocorrência de uma detecção, o sensor correspondente emite um alarme, o qual é recebido por todos sensores que estão dentro do seu raio de comunicação.

As mensagens de alarme contem uma marca temporal, a posição do nó emissor, e o tipo de ameaça detectada. Os dois primeiros componentes do alarme o identificam de maneira única, evitando duplicação de alarmes. Este trabalho considera a atomicidade de eventos reportados pelos alarmes, o que significa que cada alarme emitido indica uma ameaça distinta. Consequentemente, se a ameaça indicada é um grupo de pessoas ou veículos, eles são tratados como uma entidade única. Isto assume que sensores vizinhos cooperam para agregar informações de decisão para identificar e caracterizar ameaças, antes de emitir alarmes.

Os principais elementos deste cenário estão representados na Figura 1.4. Nesta figura ilustra-se a detecção de uma possível ameaça feita por um sensor terrestre, marcado como um círculo negro para distinguir dos demais nós sensores. Este nós emite um alarme que é recebido por todos seus vizinhos. Um destes vizinhos retransmite o alarme, o qual é então recebido por um nó vizinho perto do VANT.

Quando um alarme ocorre, um VANT, equipado com sensores mais sofisticados, é selecionado para voar na direção da área onde o alarme foi emitido. Com seus sensores mais sofisticados, o VANT é capaz de realizar a aquisição de informações mais detalhadas sobre a possível ameaça, confirmando-a ou não como ameaça, e.g. um intruso ou um foco de fogo.

1.3 Objetivos e Delimitação do Escopo

No contexto dos cenários apresentados acima, existem vários problemas a serem solucionados. O objetivo deste trabalho é prover soluções com baixo custo em termos de comunicação que deem suporte a cooperação entre os sensores estáticos e móveis objetivando a execução de missões de monitoramento. Levando em consideração as diferenças em relação à mobilidade dos sensores, este objetivo geral pode ser dividido em três outros específicos a saber:

a) Para a cooperação entre sensores estáticos, o objetivo é prover uma estratégia de distribuição de missões na rede e selecionar os sensores mais apropriados para executarem as missões enquanto a comunicação entre os sensores é reduzida para realizar essas ações de distribuição e seleção, para reduzir o consumo de energia devido à comunicação entre os nós;

b) Para a cooperação entre os nós móveis, o objetivo é prover uma solução que ajude as missões atingirem suas MAs e maximizar o tempo que elas permanecem dentro dos limites de suas MAs durante o intervalo de tempo de execução das missões, observando limitações de comunicação e energia;

c) Para a cooperação entre sensores estáticos e móveis, o objetivo é prover uma estratégia que permita a sua comunicação e seleção do sensor móvel apropriado para responder a um alarme, enquanto a comunicação entre os nós seja minimizada para poupar recursos energéticos.

Deste escopo está excluído o estudo de problemas como linguagens e abstrações para a especificação de missões (MADDEN et al., 2005), bem como suas interfaces. Fusão (NAKAMURA; LOUREIRO; FRERY, 2007) e agregação (RAJAGOPALAN; VARSHNEY, 2006) de dados também não são considerados. O escopo também exclui temas relacionados a problemas de mobilidade dos sensores, como aerodinâmica de VANTs (RAUF et al., 2011), formações de movimento conjunto (BEARD et al., 2006), sistemas anti-colisão (ALEJO et al., 2009), trabalho em times com restrições temporais (BEARD et al., 2006), trabalho em conjunto para cobertura de áreas (BEARD et al., 2006), planejamento de trajetórias (GAO 2010) e seguimento de objetos com sensores móveis (KIM; GU; POSTLETHWAITE, 2008). Segurança, que é um problema muito relevante em RSSFs (WANG; ATTEBURY; RAMAMURTHY, 2006), também está fora do escopo desta tese. Problemas de dependabilidade são considerados nas suposições para as soluções propostas, mas uma discussão mais profunda e exaustiva sobre o tema está fora do escopo da tese, mesmo considerando sua importância. Por exemplo, garantia de entrega de mensagens é um problema importante no contexto de dependabilidade que está fora do escopo da tese.

1.4 Formulação do Problema

Apresentado o cenário motivador da tese, a delimitação do seu escopo e seus objetivos, os problemas endereçados podem ser formulados.

O problema principal tratado nesta tese pode ser abstraído como o clássico problema de disseminação de dados em RSSFs (AKYILDIZ et al., 2002) (AKYILDIZ; KASIMOGLU, 2004) combinado com problema de alocação de recursos em sistemas distribuídos (TANENBAUM; STEEN, 2007). Neste problema, missões de sensoriamento podem ser vistas como tarefas e nós sensores podem ser considerados recursos disponíveis que são necessários para a execução dessas tarefas.

Em relação à disseminação de dados, o problema é como transmitir dados entre os nós sensores de maneira eficiente de tal forma que eles possam cumprir as missões. O problema de alocação de tarefas e recursos é estudado no domínio do Problema de Atribuição Ótima (GALE, 1960), e é usado, por exemplo, para definir o Problema de Alocação de Múltiplos Robôs (MRTA), como discutido em (DRESSLER, 2007), ou

mais especificamente no contexto de redes de sensores e atuadores (SANETs) como discutido em (AKYILDIZ; KASIMOGLU, 2004).

Considerando a motivação desta tese, a diferença de mobilidade entre os sensores que fazem parte da rede originam instancias distintas deste problema combinado, como apresentado a seguir.

Observando o primeiro cenário apresentado na Seção 1.2.1, da inserção da missão na rede (Figura 1.2b), através da disseminação a missão deve primeiro atingir a MA, e então informar todos os nós sensores dentro dos limites da MA. Logo, deve-se ter um mecanismo que realize este processo. Ao chegar aos nós sensores, a missão deve ser alocada a nós que possam executá-la, concluindo o setup da rede. Considerando a dinamicidade das condições de operação, como mudanças na topologia da rede devido interferências, e restrições no uso dos recursos energéticos dos nós sensores, estratégias que objetivem um setup de missão eficiente devem ser flexíveis e econômicos em termos de comunicação. Estes problemas são chamados: a) disseminação de missão de sensoriamento; e b) alocação de missão de sensoriamento, respectivamente.

Para o segundo cenário, uma rede oportunística de nós sensores móveis tem o objetivo de executar missões de sensoriamento dentro de uma MA. Dependendo da aplicação, não é obrigatório que os nós sensores executem apenas essas determinadas missões. Aproveitando a característica oportunística da rede formada pelos nós móveis, eles podem ser “contratados” para executar missões quando a direção de seu movimento está de acordo com a MA de uma determinada missão. O problema de disseminação de missão neste cenário é como transferir missões entre os nós sensores de tal forma que elas alcancem e permaneçam em nós sensores localizados nas suas respectivas MAs. A parte do problema relacionada a alocação de missão está entrelaçada com a disseminação, uma vez que a missão ao atingir um nó que tem como destino a sua MA, ou ao menos movendo-se na direção de sua MA, a missão é considerada alocada ao nó.

Na situação descrita no terceiro cenário, o primeiro problema a ser tratado é como fazer as mensagens emitidas pelos sensores estáticos alcançarem os móveis de maneira eficiente. Em seguida, considerando-se nós móveis com diferentes capacidades, a escolha de um que seja apropriado para responder a requisição feita pelos nós estáticos deve ser tratada. Novamente, a preocupação com o consumo de energia deve ser considerada. Mesmo que isto não seja um problema para os nós móveis, os nós estáticos não gastar muita energia para interagir com os outros nós. O primeiro problema chama-se entrega do alarme, uma vez que as requisições dos nós estáticos são consideradas como alarmes por eles emitidos, enquanto o segundo é chamado de designação do tratador do alarme.

Para todos os cenários considerados assume-se que para reduzir a comunicação entre os nós deve-se usar mecanismos que diminuam o número de mensagens por eles enviadas. Esta suposição desconsidera problemas de comunicação que poderiam afetar a recepção das mensagens, e por isto iriam requerer esquemas de retransmissão, afetando o número de mensagens que são enviadas e recebidas.

Depois desta breve introdução que identifica cada problema tratado no trabalho, eles são discutidos em detalhes nas seções a seguir.

1.4.1 Disseminação e Alocação de Missões de Sensoriamento em RSSFs Estáticas

A fase de configuração de uma RSSF apresenta problemas relacionados à como informar os nós sensores a respeito da missão de sensoriamento a ser realizada e também a questão de selecionar quais sensores devem executá-la. É importante reparar a suposição de que missões submetidas à rede são possíveis de serem tratadas pelos sensores que a compõe, i.e. a rede tem os recursos necessários para executar as missões.

Ademais, considerando-se ambientes dinâmicos nos quais mudanças afetam as capacidades dos sensores para o sensoriamento e para realizar tarefas de rede, eles têm que ser capazes de adaptar seus comportamentos de forma que estejam preparados para aceitar novas missões. Logo, os problemas a serem endereçados são:

a) Disseminação de missão de sensoriamento;

b) Alocação de missão de sensoriamento.

O primeiro problema é como realizar a disseminação de uma missão entre os nós de uma rede, i.e. como fazer com que os nó sensores tenham conhecimento da missão, de forma eficiente. Uma solução trivial é realizar a difusão da missão para todos os nós através de inundação, mas esta não é uma alternativa eficiente, considerando-se que nem toda informação referentes a missões é interessante a todos os nós sensores.

O segundo problema é como dividir a carga de trabalho entre os nós depois que eles recebem uma nova missão. Uma solução trivial para este problema de alocação de missão seria tomar decisões centralizadas e enviar partes específicas das missões para nós específicos que terão que tratá-las. No entanto, um nó decisor central, com uma visão global da rede, deve ter informações sobre todo o contexto de operação, sobre a rede e sobre as condições do ambiente, de modo a funcionar adequadamente. Isto requer um desnecessário tráfego de mensagens de controle vindo de toda a rede para este nó centralizador, logo usando de maneira não eficiente recursos de comunicação, além de efeitos com atrasos decorrentes desse tráfego desnecessário. Além disto, um nó decisor central representa um ponto único de falha, o que afeta de maneira negativa a confiabilidade de todo o sistema. Por outro lado, a decentralizando a decisão da alocação de missão, sendo feita de maneira autônoma pelas nós e de modo dinâmico, pode gerar resultados melhores dentro de restrições de tempo e de comunicação, além de aumentar a confiabilidade do sistema.

Uma questão que faz parte da segunda parte do problema se refere a como os nós devem adaptar seus comportamentos para realizar a divisão da carga de trabalho, baseados nas condições de operação da rede. Isto é importante porque estas condições podem influenciar a maneira pela qual a missão deve ser dividida. Se o sistema precisa esperar intervenções de operadores, talvez se torne muito lento e, portanto, a capacidade de se tomar decisões autonomamente em cada nó pode providenciar benefícios no sentido de se alocar novas missões a nós sensores mais convenientes. Ademais, o sistema deve ser robusto para enfrentar situações adversas, como presença de “hotspots” e distribuição de nós ineficiente, o que pode comprometer os resultados finais.

Ortogonalmente aos problemas mencionados acima, existe a preocupação com a utilização da comunicação entre os nós. Como as atividades executadas pelos sensores para distribuir as missões e para informar outros nós sobre suas características e condições de operação utilizam comunicação, esta utilização está fortemente relacionada aos problemas discutidos acima e suas possíveis soluções. Além disto,

tendo em vista que a comunicação é uma das atividades mais caras do sob a ótica do consumo de energia (MINI; LOUREIRO, 2009), essas atividades impactam o consume de energia, o que é uma grande preocupação em RSSFs devido ao limitado recurso de energia que os nós sensores dispõem (AKYILDIZ et al., 2002).

1.4.2 Disseminação de Missões de Sensoriamento em RSSFs Móveis

Considerando o segundo cenário, espera-se que várias missões diferentes sejam executadas em diferentes MAs, como a MA-2, dentro de uma área coberta por um sistema de monitoramento. Nesta área coberta pelo sistema, diferentes tipos de nós sensores móveis podem estar se movendo de acordo com diferentes tipos de padrão de movimento, e várias missões de sensoriamento podem ter sido submetidas ao sistema. Cada missão se refere a uma específica MA parte da área total coberta pelo sistema.

Assume-se que todos os sensores móveis que fazem parte do sistema têm as capacidades de processamento e sensoriamento necessárias para executar as missões. Além disto, eles estão necessariamente restritos à execução de nenhuma missão em particular, mas espera-se que eles cooperem e executem qualquer missão quando solicitados. No entanto, diferentemente do problema de disseminação e alocação apresentado anteriormente, o qual considera apenas sensores estáticos, no segundo cenário assume-se apenas a presença de sensores móveis. Neste caso, o problema de disseminação é diferente, uma vez que ela não se resolve como no anterior, mas continua por todo o tempo em que a missão deve ser executada até o seu término, uma vez que os nós sensores não permanecem na MA, mas se movem pela por toda área entrando e saindo da MA várias vezes. Os nós então tem que ser capazes de transferir as missões para outros nós de acordo com suas posições atuais e seus destinos. Além disto, como se assume que todos os sensores têm as capacidades necessárias para realizar as missões, a condição que os elegem para executá-las é então a sua posição coincidir com a localização da MA, logo não existe problema de alocação com apresentado na subseção anterior, uma vez que a disseminação por si só já aloca a missão ao nó que a recebeu.

Do ponto de vista da missão, é importante que ela permaneça dentro dos limites de sua MA. Logo, o problema de disseminação de missão a ser endereçado se refere a uma maneira de se realizar a transferência da missão de forma eficiente de um nó localizado fora de sua MA para outro de modo que ela chegue a um nó dentro de sua MA. A razão para este comportamento é tentar maximizar o tempo no qual as missões permanecem dentro de suas MAs durante o intervalo de tempo no qual elas devem ser executadas. A situação apresentada na Figura 1.3 é um caso particular no qual um nó que está hospedando a missão e está deixando a sua MA, S-1, encontra outro nó que está entrando na MA, S-2. Além da questão da densidade de nós mencionada na Seção 1.2.2, sob uma perspectiva mais perspectiva na qual se considera uma área com um número maior de nós e MAs, considerando-se a inserção de novas missões na rede, o problema é como selecionar nó sensores móveis de uma forma eficiente. Exemplos de questões que surgem neste cenário são: vale a pena migrar uma missão para qualquer nó com o qual se encontra, ou deve-se considerar a direção do movimento dos nós de modo a se minimizar o número de transferências de missões?

Transferências desnecessárias são aquelas que não levam nem aproximam as missões de suas MAs. Tais transferências apenas mantem a situação corrente de uma missão, na qual ela se encontra fora de sua MA, ou faz com que a situação ainda piore, i.e. leva a missão para ainda mais longe de sua MA. Este último caso é obviamente indesejável do ponto de vista funcional do sistema. O primeiro tipo de transferência desnecessária é também indesejável, mas devido a outros aspectos, como sigilo (LEE et al., 2009b), instabilidade de enlaces de comunicação (SOLTANI; MISRA; RADHA, 2008), e preservação de energia (TEI et al., 2005) (nos casos em que a questão de preservação de energia são considerados).

A partir das observações acima, o problema de disseminação de missões em RSSFs móveis pode ser resumido em como se executa a transferência de missões entre nós de modo que elas consigam chegar às suas respectivas MAs, e como maximizar o tempo no qual as missões são hospedadas por nós dentro das MAs, enquanto o número de transferências é minimizado.

1.4.3 Tratamento Alarmes em RSSFs Estáticas e Móveis Cooperativas

A cooperação e interação entre nós sensores estáticos e móveis em RSSFs é fortemente influenciada por restrições de energia e comunicação. Neste terceiro cenário, considera-se que os nós estáticos sofrem de severas restrições de energia, como já discutido.

Nós sensores móveis, VANTs de pequeno porte, são considerados neste cenário, e portanto não são capazes de levar cargas tão pesadas quanto plataformas VANTs de grande porte, e isto afeta diretamente suas capacidades de comunicação e alcance. Outra restrição ligada com a questão da carga é que plataformas VANT de pequeno porte devem usar sua energia de modo eficiente, uma vez que eles não são capazes nem de levar muito combustível nem baterias grandes. Este fato impacta não apenas o subsistema de comunicação, mas também restringe o alcance de utilização operacional de VANTs de pequeno porte, limitando suas possibilidades de cooperação. Logo, mesmo que os sensores móveis não tenham restrições tão severas de energia como os sensores estáticos no solo, eles devem utilizar seus recursos energéticos de modo cauteloso.

Para se combinar ambos os tipos de sensores e fazer com que eles trabalhem cooperativamente, comunicação é um aspecto de grande importância. No entanto, o problema se refere a como localizar os nós móveis para que as mensagens enviadas dos nós estáticos sejam entregues aos móveis. Além disto, como selecionar nós móveis que são mais adequados a responder um determinado alarme, considerando nós móveis com diferentes capacidades. Logo, estes problemas podem ser definidos como segue:

a) Entrega de Alarme: como entregar ou rotear alarmes de maneira eficiente dos nós estáticos aos móveis (VANTs), e;

b) Designação de Tratador de Alarme: como decidir a designação de um nó sensor móvel (VANT) para tratar um determinado alarme.

O primeiro problema pode ser tratado pelo menos de duas maneiras: 1) através de entidade central que reúne informações sobre todos os alarmes e então os distribuem entre os nós móveis; ou 2) através de um tratamento de informação decentralizado e um processo distribuído que direciona os alarmes diretamente para os nós sensores móveis

via colaboração entre os nós estáticos para reencaminhar os alarmes. Cada opção leva a diferentes efeitos. No entanto, considerando que a solução centralizada dificilmente escalaria com o aumento no número de nós estáticos, nós móveis e alarmes, e que seria esperado altos custos em termos de comunicação, a solução decentralizada parece mais razoável (MUTAMBARA, 1998). Logo, o problema (a) é como providenciar um mecanismo que realize a entrega de alarmes de forma eficiente sem um nó coordenador central.

O segundo problema (b) se refere a decidir se um nó sensor móvel que está disponível é adequado ou não para tratar um determinado alarme. Esta decisão deve considerar as características de cada nó sensor móvel acessível e a análise da ameaça que disparou o alarme, bem como qual o tipo de sensor que é necessário para realizar a confirmação. Este segundo problema também afeta o primeiro, uma vez que pode impactar a maneira como a entrega de alarmes é realizada.

1.5 Abordagens Propostas

A tese propõe o uso de diferentes técnicas de agentes de software para resolver os problemas identificados. Dentre elas, a técnica mais importante é a de multi-agentes utilizando agentes de software móveis e algoritmos com comportamento biologicamente inspirado. Essas técnicas foram escolhidas primeiramente porque abordagens multi-agente fornecem modelos naturalmente aplicáveis a sistemas altamente distribuídos, como RSSF, com abstrações e protocolos para cooperação decentralizada entre entidades independentes (WOOLDRIDGE, 2002). A escolha por agentes móveis se baseia no fato de que eles fornecem flexibilidade para (re)instalar software na rede, além do fato de possibilitar a combinação de dados e inteligência (código)nas mensagens comunicadas no sistema (WOOLDRIDGE; JENNINGS, 2002). Algoritmos biologicamente inspirados fornecem soluções simples e naturalmente decentralizadas com características desejáveis em RSSFs, tais como auto-organização (DRESSLER; AKAN, 2010).

1.5.1 Disseminação e Alocação de Missão de Sensoriamento em RSSFs Estáticas

Para os problemas relacionados à configuração de sensores estáticos, necessários para se uma determinada missão, a abordagem proposta neste trabalho usa sistemas multi-agentes (WOOLDRIDGE, 2002) em que agentes de software móveis disseminam missões de sensoriamento entre os sensores e decidem sobre a sua alocação, através de cooperação com agentes estacionários nos sensores. A idéia se baseia no fato de que decisões locais podem ser tomadas pelos agentes nos nós sensores, evitando a troca desnecessária de dados entre os sensores e a necessidade de um ou vários nós coordenadores da rede. Evitando a troca dessas mensagens, o total de mensagens comunicadas na rede diminui, e, por conseguinte, o consumo de energia. Para enfatizar a importância da compensação entre comunicação e computação local, vale mencionar uma “regra de ouro” que vale em muitos casos para RSSFs, que afirma que a transmissão de um 1 bit corresponde a execução de 1000 instruções em termos de consumo de energia (HILL, 2003).

No entanto, existe um limite para redução da comunicação entre os nós. Este limite se refere à comunicação necessária para garantir que os nós recebam informação sobre as missões, i.e. o que a rede deve fornecer de informação como resultado de uma missão. Por outro lado, em abordagens tradicionais, nós sensores recebem comandos específicos sobre o que eles devem fazer e quando, o que requer muita comunicação entre nós centrais da rede, aqueles que distribuem as tarefas, e os nós sensores (HONG et al., 2008). Isto se deve ao fato de que para particionar e alocar as tarefas para os nós mais adequados, os nós coordenadores centrais devem periódica, ou esporadicamente, coletar dados sobre o status de toda a rede. Isto representa um custo adicional em termos de tráfego de dados e, portanto, consumo de energia, além de outros problemas, como a existência de um ponto único de falha. Por outro lado, novas abordagens tentam fazer os nós o mais autônomos possível, fazendo com que sejam capazes de tomar decisões sobre o que devem executar baseados apenas em dados gerais que descrevem as missões, i.e. especificação das missões. Mesmo demandando uma troca adicional de dados, dependendo da solução proposta, este tipo de abordagem é capaz de explorar o limite mínimo de comunicação entre os nós, para reduzir o total de tráfego na rede correspondente a mensagens de controle, e assim economizar energia de maneira significante (HEIMFARTH et al., 2010a). Abordagens baseadas em agentes para RSSFs fazem parte de um novo tipo de estratégias que fazem com que os nós sensores tenham comportamento autônomo (TYNAN; O'HARE; RUZZELLI, 2006). Como esta motivação, adota-se neste trabalho uma abordagem multi-agente para tratar o problema da disseminação e alocação de tarefas.

O mecanismo de raciocínio usado pelos agentes de software para a tomada de decisões para executar as missões de disseminação é baseado num procedimento de decisão probabilístico que explora informações locais sem a necessidade de comunicação adicional entre os nós. Além dele, um mecanismo biologicamente inspirado baseado no comportamento de abelhas também é adotado para tornar a solução de alocação de missões mais robusta. Esta abordagem imita o comportamento das abelhas quando procurando comida na natureza, e esta analogia é explorada por um dos tipos de agentes de software móveis que faz parte da abordagem multi-agente para distribuir informações entre os nós sensores.

1.5.2 Disseminação de Missões de Sensoriamento em RSSFs Móveis

Como discutido na Seção 1.4.2, a efetividade da execução oportunística das missões de sensoriamento pelos sensores móveis enquanto eles se movem pelas áreas de monitoramento em direção a suas MAs depende do quão precisa e eficientemente as missões chegam e permanecem em suas MAs. Esta eficiência também depende do quão bem as missões são transferidas de um nó sensor para outro.

Missões de sensoriamento são caracterizadas por requisições de medições de fenômenos físicos que devem ser executadas dentro de um intervalo de tempo numa certa área. Logo, elas podem ser vistas como serviços que devem ser executados na plataforma fornecida pelos nós sensores, utilizando seus recursos de sensoriamento e processamento. Além disto, a decisão de manter a missão em um nó ou transferi-la para outro deve variar de acordo com diferentes critérios. No cenário analisado, este critério é relacionado a MA, mas ele pode ser qualquer outro, e diferente para diferentes

missões. Sendo assim, existe uma forte ligação entre a missão e o controle sobre sua transferência entre os nós sensores.

Agentes de software móveis fornecem uma abordagem modular para implementar serviços que proveem comportamento inteligente que permite a tomada de decisões baseadas em critérios específicos, os quais podem ser relacionados ao seu próprio movimento (LANGE; OSHIMA, 1999). Esta característica apresenta uma correspondência perfeita entre a descrição das missões apresentadas acima e o que agentes de software móveis podem oferecer. Sendo assim, esta tese apresenta uma abordagem multi-agente na qual agentes de software móveis são usados para implementar a disseminação de missões de sensoriamento em uma RSSFs móvel. Uma missão é codificada, encapsulada e carregada por um agente de software, o qual tem a capacidade de raciocinar se deve se transferir para outro nó sensor seguindo as diretivas da missão.

A despeito desta correspondência entre o modelo de agentes de software móveis e as necessidades identificadas, um elemento adicional é necessário para o sucesso desta abordagem. Este elemento é o uso de informações de contexto para dar suporte às decisões dos agentes. Logo, uma abordagem sensível ao contexto é adotada para fazer capacitar os agentes a usar informações de contexto de forma que decisões mais apropriadas para as suas transferências entre os nós sejam tomadas, i.e. decisões sobre se os agentes devem permanecer num nó ou migrar para outro. Este trabalho investiga diferentes variantes desta abordagem (representado por diferentes níveis de inteligência) na forma como se usa a informação de localização que suporta as decisões dos agentes para maximizar sua estada em sua respectiva MA. Como esta é uma proposta geral baseada no cenário apresentado na Seção 1.2.2, deve-se considerar sensores de diferentes tipos, incluindo aqueles que têm restrições de recursos. Sendo assim, restrições de energia devem ser consideradas.

Este problema motiva a afirmação sobre a minimização do número de transferências de missões entre os nós apresentada na Seção 1.4.2, que é endereçada neste trabalho.

A abordagem proposta objetiva aumentar a eficiência das migrações dos agentes entre os nós sensores móveis, i.e. aumentando a sua estada dentro de suas respectivas MAs e minimizando o número de migrações entre os nós. Nesta solução não se considera a coleta de dados pelos sensores, o que pode ser endereçado por soluções como a apresentada em (LEE et al., 2009b), na qual um agente coletor realiza esta tarefa.

1.5.3 Tratamento de Alarmes em RSSFs Cooperativas Estáticas e Móveis

A combinação de ambos sensores estáticos e móveis representa uma solução promissora para melhorar os resultados obtidos por uma RSSF em aplicações de monitoramento. A despeito dos resultados promissores, a utilização conjunto desses dois tipos de sensores e redes apresenta problemas que precisam ser resolvidos para fazer o sistema funcionar corretamente, como discutido na Seção 1.4.3. A cooperação entre os dois tipos de nós apresenta maior complexidade se comparado a RSSFs tradicionais, compostas apenas por nós estáticos ou móveis. Esta complexidade adicional é explicada por diversas questões que variam desde a utilização eficiente de energia pelos nós sensores com restrição de recursos até o emprego eficiente dos nós

sensores móveis durante a operação do sistema, através do seu deslocamento para lugares onde eles são necessários.

O mecanismo proposto para endereçar os problemas objetiva realizar uma comunicação eficiente entre os nós, minimizando o número de mensagens trocadas na rede, logo também diminuindo o custo em termos de consumo de energia. Para alcançar este objetivo, uma abordagem biologicamente inspirada foi formulada, a qual explora os conceitos de feromônios artificiais e estigmergia (BONABEAU; DORIGO; THERAULAZ, 1999) como meio de disseminar informações necessárias para fazer os nós sensores cooperarem. Para endereçar o primeiro problema descrito na Seção 1.4.3, o objetivo dos feromônios artificiais deixados pelos nós móveis é fornecer informação sobre sua localização atual aos sensores estáticos. Esta solução é então ampliada para considerar sensores móveis com características diferentes, o que é representado por diferenças entre o aroma dos feromônios deixados por eles. Este melhoramento endereça o segundo problema, permitindo a escolha de sensores móveis apropriados para responder aos alarmes.

1.6 Contribuições

De acordo com os problemas endereçados e os objetivos da tese, baseado nas abordagens adotadas, as principais contribuições apresentadas são:

a) ***Disseminação e Alocação de Missões em RSSFs Estáticas*** - fornecer suporte para a configuração de RSSFs através de decisões decentralizadas sobre a disseminação e alocação de missões de sensoriamento, inserindo agentes móveis inteligentes nos nós sensores.

Esta contribuição consiste na proposta e avaliação experimental de uma abordagem multi-agente na qual agentes de software móveis são capazes de levar missões de sensoriamento aos nós sensores. Através do uso de informações locais adquiridas dos nós sensores em uma determinada vizinhança, os agentes são capazes de tomar decisões sobre a alocação dos nós para executar as missões. Desta forma, esta proposta contribui com a economia de energia, através da redução da comunicação entre os nós sensores. Este efeito é alcançado primeiramente por evitar soluções convencionais de inundação de mensagens na rede para a disseminação de missões, e por realizando a tomada de decisão localmente, evitando a comunicação de mensagens adicionais entre os nós. Resultados experimentais indicam economia significativa no número de mensagens trocadas entre os nós comparado a estratégias convencionais baseadas em inundação. Esta contribuição for primeiramente publicada em (FREITAS et al., 2009e), (FREITAS et al., 2009b). Uma análise mais detalhada foi apresentada em (FREITAS et al., 2009c), (HEIMFARTH et al., 2010b) e (FREITAS et al., 2011c), que baseado o melhoramento da proposta apresentado em (FREITAS et al., 2010b). Em (FREITAS et al., 2011d) a contribuição é resumida.

b) ***Disseminação de Missões em RSSFs Móveis*** – propõe a instalação de serviços de redes de sensores através da disseminação de missões de sensoriamento para nós sensores móveis apropriados que não sejam exclusivamente dedicados à execução de tais missões, mas que podem ser utilizados de maneira oportunística para tal.

Esta parte do trabalho investiga uma abordagem baseada no uso de agentes de software móveis que implementam missões de sensoriamento que podem migrar entre nós sensores de acordo com a posição geográfica dos nós. A idéia é configurar uma "rede de sensores virtuais", implementada pelos agentes de software que são executados nos nós sensores. O objetivo é manter os agentes hospedados nos nós que estão localizados nas áreas de interesse para uma determinada missão que é executada pelos agentes. Esta proposta inclui o uso de informações geográficas de contexto para dar suporte aos mecanismos de decisão usados pelos agentes móveis, explorando, testando e comparando diferentes estratégias. Esta contribuição foi primeiramente apresentada em (FREITAS et al., 2010d) sendo depois estendida e melhor analisada em (FREITAS et al., 2011a).

c) ***Entrega e Designação de Tratadores de Alarmes*** – propõe e investiga maneiras de realizar a utilização cooperative de sensores móveis e estáticos em missões de monitoramento de área.

O maior desafio para uma efetiva cooperação entre nós estáticos e móveis está principalmente relacionado à como fornecer informações aos nós estáticos sobre a posição atual dos sensores móveis mais adequados. Este trabalho fornece uma contribuição relacionada a este tópico, apresentando uma abordagem para rotear e entregar mensagens dos nós estáticos aos móveis. Esta abordagem usa conceitos biologicamente inspirados e é implementada através de feromônios artificiais que primeiramente ajudam a achar e então indicam a direção de movimento dos nós móveis. Esta informação é usada para rotear mensagens de alarme endereçadas aos nós móveis. Além disto, a proposta é estendida e melhorada para escolher o nó sensor móvel mais apropriado para tratar um determinado alarme. Vários experimentos foram realizados para explorar diferentes cenários e características da solução proposta, ressaltando e discutindo vários aspectos relevantes. Esta contribuição foi primeiramente apresentada em (FREITAS et al., 2009a) e (FREITAS et al., 2009b), tendo os primeiros resultados experimentais publicados em (FREITAS et al., 2009d). Maiores detalhes da proposta são apresentados em (FREITAS et al., 2010a), enquanto mais experimentos e analises mais detalhadas são apresentados em (FREITAS et al., 2010c) e (FREITAS et al., 2010e). Um resumo da proposta é apresentado em (FREITAS et al., 2011e).

Transversal às principais contribuições listadas encontra-se a preocupação com o aspecto de consumo de energia. Todas as contribuições representam esforços para diminuir a comunicação entre os nós sensores. Desta forma, elas devem ser entendidas no contexto contribuir para a redução no consumo de energia.

1.7 Metodologia

Um conjunto de cenários é usado como base experimental para o estudo dos problemas relacionados ao uso de RSSFs para monitoramento. A especificação detalhada desses cenários estabelece o escopo e os limites dos experimentos. Experimentos de pequena escala são usados para entender e testar a adequação das abordagens propostas para resolver os problemas. Os resultados desses experimentos suportam a modelagem de experimentos de maior escala para validação das soluções propostas.

A metodologia para condução desses experimentos, bem como para a análise de seus resultados, segue as orientações para realização de experimentos aleatorizados descritas em (BOX; HUNTER; HUNTER, 2005).Essas orientações explicam como as variáveis estudadas devem ser controladas para construção de experimentos em bloco, de forma que seja possível observar variações de interesse em experimentos aleatorizados bem como como analisar e interpretar os resultados alcançados. Experimentos aleatorizados são usados devido ao grande número de possibilidades geradas pelos cenários considerados nesta tese, por exemplo, áreas de diferentes dimensões, diferentes números possíveis de nós, diferentes posicionamentos desses nós nas áreas e padrões de movimento. Sendo assim, o uso desta metodologia torna possível alcançar resultados razoáveis com uma cobertura representativa dos casos, bem como apresentando resultados sem viés, e que, portanto podem ser usados para basear conclusões sobre o comportamento geral das soluções propostas aplicadas aos cenários considerados.

A comparação entre os resultados obtidos pelas soluções propostas e soluções de referência é o método adotado para se avaliar objetivamente a qualidade das contribuições. Uma comparação subjetiva com trabalhos relacionados também é apresentada. Esta comparação tem que ser subjetiva, uma vez que nenhum dos trabalhos relacionados fornece resultados experimentais que sejam diretamente comparáveis aos obtidos.

O principal método para a validação experimental se dá por meio de simulação. A motivação para esta escolha por simulações computacionais tem duas razões principais. A primeira se refere ao fato de que é muito mais barato a construção de simulações de larga escala, com centenas ou mesmo milhares de nós, comparado à implementação de protótipos físicos. A segunda se relaciona ao fato de que hardware e software específicos usados em demonstradores geralmente consomem um tempo significante para sua correta configuração para que sejam usados na avalição que deve ser realizada sobre o comportamento da rede quando executando as soluções propostas; isto poderia comprometer o progresso do trabalho, e possivelmente levar a uma perda de foco. Adicionalmente, o uso de modelos analíticos também não fornece uma boa alternativa, devido à alta complexidade dos cenários estudados. Modelos analíticos para tais cenários seriam muito simplificados para serem tratáveis, o que diminuiria consideravelmente sua capacidade de expressar os aspectos mais relevantes que são possíveis de serem manipulados com simulações.

A escolha da ferramenta de simulação leva em conta peculiaridades dos tipos de nós sensores usados em RSSFs foco deste trabalho (redes compostas de nós com diferentes capacidades de movimento). Isto motivou o estudo comparativo de diferentes ferramentas e suas características, bem como as possibilidades que elas oferecem (SINGH; VYAS; TIWARI, 2008) (LESSMANN et al. 2008). A comparação indicou que nenhuma ferramenta oferecia todas as características necessárias para o estudo das soluções propostas no trabalho. Os principais pontos considerados são: facilidade para realizar simulações de redes sem fio; fornecimento de modelos de movimento; usabilidade e mecanismos de extensão amigáveis; suporte a visualização de resultados; e código aberto que possibilite a modificação da ferramenta de acordo com as necessidades. Levando em conta esses critérios, uma ferramenta de simulação especialmente desenvolvida para redes sem fio e utilizada com sucesso para RSSFs foi escolhida. Esta ferramenta se chama ShoX (LESSMANN; HEIMFARTH; JANACIK,

2008) que foi originalmente desenvolvida na Universidade de Paderborn. Através da extensão desta ferramenta com características necessárias para o teste das soluções propostas nesta tese, como a possibilidade de se ter nós com diferentes modelos de movimento numa mesma simulação, uma nova ferramenta foi criada chamada de GrubiX (HEIMFARTH; FREITAS, 2011).

Adicionalmente as simulações, um demonstrador de pequeno porte foi desenvolvido implementando um dos algoritmos apresentados na tese. Este demonstrador não tem o mesmo objetivo das simulações, mas tão somente objetiva aferir a viabilidade da abordagem proposta usando software e hardware disponível nos dias de hoje.

1.8 Estrutura da Tese

A tese apresenta nove capítulos além da introdução. O Capítulo 2 fornece informações sobre RSSFs, no qual uma visão geral dos principais problemas nesta área de pesquisa é mencionada e uma descrição resumida de abordagens tradicionais para trata-los é apresentada. Além disto, como agentes de software móveis e abordagens biologicamente inspiradas são usadas nas soluções propostas neste trabalho, o Capítulo 2 também inclui uma breve descrição de conceitos básicos e técnicas que são usadas nestes dois tipos de abordagens.

O Capítulo 3 descreve a solução proposta para a configuração de RSSFs estáticas. Uma visão geral da solução aos problemas apresentados na Seção 1.4.1 é apresentada, seguido da definição das missões de sensoriamento e das suposições consideradas. Na sequência apresentam-se os detalhes de cada parte da solução, e finalmente os resultados experimentais e discussões sobre eles concluem o capítulo.

O Capítulo 4 descreve a solução baseada em agentes móveis para a disseminação de missões entre os nós sensores móveis, para endereçar o problema descrito na Seção 1.4.2. Diferentes níveis de inteligência que suportam a decisão dos agentes para realizar migrações são explorados na solução proposta, tendo os resultados obtidos por elas comparados entre si.

O Capítulo 5 apresenta a solução proposta para o uso cooperativo de nós sensores estáticos e móveis. A primeira parte do capítulo traz as definições e importantes considerações sobre as suposições assumidas, seguido por uma descrição da solução baseada em feromônios artificiais para realizar a entrega de alarmes enviados dos sensores estáticos para os móveis. Melhorias nas características da solução proposta são apresentadas de maneira incremental. Em seguida uma análise de viabilidade é apresentada, na qual a solução é analiticamente testada para garantir sua viabilidade considerando condições realísticas. O capítulo é concluído com a apresentação e discussão dos resultados experimentais obtidos.

O Capítulo 6 apresenta detalhes sobre o simulador GrubiX, e como as simulações usadas para testar as soluções proposta foram implementadas nesta ferramenta.

O Capítulo 7 apresenta o demonstrador protótipo que foi desenvolvido para avaliar a viabilidade da proposta do trabalho, implementando um dos algoritmos desenvolvidos em uma RSSFs física.

O Capítulo 8 discute brevemente aspectos relevantes sobre dependabilidade em RSSFs, os quais podem comprometer o desempenho das soluções propostas neste

trabalho. Como aspectos relacionados a dependabilidade não estão incluídos nos objetivos da tese, a intensão é apenas fornecer uma visão geral de tais aspectos, afirmando a consciência do autor em relação a eles, mesmo que eles não tenham sido tratados no escopo da tese.

O Capítulo 9 discute trabalhos relacionados a esta tese, realçando alguns aspectos similares bem como diferenças entre esta tese e os trabalhos analisados.

O Capítulo 10 fornece um sumário da tese trazendo as conclusões de cada contribuição. Finalmente, uma discussão sobre idéias de trabalhos futuros baseados nos resultados obtidos pela tese é apresentada.

APPENDIX D NOTE ON PUBLICATION

This thesis was developed under cooperation between the Informatics Institute at Federal University of Rio Grande do Sul and the School of Information Science, Computer and electrical Engineering at Halmstad University according a regulatory agreement signed between the vice-chancellors of both universities.

The thesis was defended and approved in both universities, as well as it is published and stored in their libraries according to their specific regulations. This copy published at Federal University of Rio Grande do Sul has the same content as the one published at Halmstad University under the ISBN 978-91-87045-00-4, with minor changes after comments from the evaluation committee at UFRGS and formatting arrangements.